FACILE GUIDA AL COMMODORE 64

SYBEX

Joseph Kascmer

EDIZIONE ITALIANA

GRUPPO EDITORIALE JACKSON

FACILE GUIDA AL COMMODORE 64

di
JOSEPH KASCMER

GRUPPO
EDITORIALE
JACKSON
Via Rosellini, 12
20124 Milano

© Copyright per l'edizione originale Sybex Inc. 1983
© Copyright per l'edizione italiana Sybex Inc. 1984

Il Gruppo Editoriale Jackson ringrazia per il prezioso lavoro svolto nella stesura dell'edizione italiana la signora Francesca Di Fiore, e l'Ing. Roberto Pancaldi.

Tutti i diritti sono riservati. Stampato in Italia. Nessuna parte di questo libro può essere riprodotta, memorizzata in sistemi di archivio, o trasmessa in qualsiasi forma o mezzo, elettronico, meccanico, fotocopia, registrazione o altri senza la preventiva autorizzazione scritta dell'editore.

Stampato in Italia da:
S.p.A. Alberto Matarelli — Milano — Stabilimento Grafico

Fotocomposizione:
CorpoNove s.n.c. — Bergamo — via Borfuro 14/c — Tel. 22.33.65

SOMMARIO

RINGRAZIAMENTI V

PREFAZIONE VII

CAPITOLO 1 — GIUNGERE AL POTERE 1
Padrone e servitore si incontrano 1
Installazione del calcolatore 1
Controllo della tastiera 4

CAPITOLO 2 — SCRIVERE SUL VIDEO 11
Controllo dello schermo 11
Stampare sul video 14

CAPITOLO 3 — PIANI E PROGRAMMI 21
Comandi di schedulazione 21
Sostituti dei numeri 24
Proiezioni numeriche 27
Comandi raggruppati 29

CAPITOLO 4 — CAPACITÀ DI DECISIONE DEL CALCOLATORE . . 31
Usare discernimento 31
Simulazioni 33

CAPITOLO 5 — CONTROLLO DI PROGRAMMA 37
Un corredo di comando per il programmatore 37
Comandi per muoversi tra le istruzioni 42
Ripetizioni limitate 45

CAPITOLO 6 — GESTIONE DI PAROLE E INFORMAZIONI 53
Stringhe: parole dal calcolatore 53
Informazioni interne 59
L'orologio interno 66
Contatto esterno 67

CAPITOLO 7 — MEMORIA DI BASSO COSTO — NASTRI 75
Uso di programmi con cassette magnetiche 75
Fatti e file su nastro 78

CAPITOLO 8 — MEMORIA-VELOCE-DISCHI 81
Un matrimonio di macchine 81
Uso di dischi preregistrati 84
Programmi dal calcolatore al disco 90
Usare più di un'unità-disco 94
Fatti e file su disco 96

CAPITOLO 9 — FARE E RIFARE I PROGRAMMI 99
Aggiungere parti di programma 99
Ricostruzione di programmi presi a prestito 105

CAPITOLO 10 — LA STORIA NASCOSTA 115
Un passo oltre l'assemblatore 115
Cos'è il Commodore 64? 119
Come espandere il calcolatore 120

APPENDICE A — COMANDI SPECIALI 123

APPENDICE B — GUIDA ALLE FRASI IN GERGO 127

RINGRAZIAMENTI

Ringrazio James Compton e lo staff editoriale della Sybex, così validi nel trasformare in libro il mio manoscritto. Un riconoscimento speciale a Heidi Miller che ha preparato l'indice e a Bill Mlotok che con la sua esperienza, capacità tecnica e buon senso ha fatto una valida revisione del testo.

Grazie anche a Paul Losness del PCP Computers di El Cerrito, CA, che mi ha fornito molte informazioni.

La poesia di pag. 13 è di Philiph Bailey, da **Festus.** I versi di pagg. 13-15 sono tratti da "La legge dello Yukon" di Robert W. Service.

PREFAZIONE

Questo libro vi insegna a usare il vostro personal computer, e scoprirete come ottenere il controllo del calcolatore in pochi giorni. A questo scopo non servono il linguaggio e la teoria della scienza dei computer, e in questo libro non ci faremo appesantire da tali elementi. Per comandare il calcolatore come se fosse un'estensione della vostra mente non occorre una speciale conoscenza della matematica o della programmazione. È infatti possibile controllare un calcolatore con la stessa facilità con cui si comanda un'automobile o una macchina da scrivere. Una volta imparate le cose essenziali, l'operazione diventa più semplice.

Se volete adoperare il Commodore 64 per un uso particolare, o se volete scoprire come controllare un personal computer, questo libro fa per voi.

Poichè potete usare un calcolatore in molti modi diversi, potete avvicinarvi a questo libro in altrettanti modi. Il **Capitolo 1** è una breve introduzione che vi dice dove sono i comandi, come collegare il vostro calcolatore e quali tasti premere o non. Qualunque sia lo scopo per il quale pensate di usare il vostro calcolatore, una scorsa a questo capitolo vi farà entrare in confidenza con l'apparecchio che sta di fronte a voi. Se pensate di usare solo programmi preregistrati potete saltare al **Capitolo 7** se i programmi sono su nastro, o al **Capitolo 8** se sono su disco; se volete fare un uso completo dei programmi su disco, leggete la seconda parte del **Capitolo 9.** Poichè però il Commodore 64 non dispone ancora di software come altri personal computer sul mercato da più tempo, non potete limitarvi a questi capitoli.

Se desiderate scrivere le vostre istruzioni sul calcolatore, dovete leggere i **Capitoli 2, 3, 4, 5 e 6** nonchè la prima parte del **Capitolo 9.**

Il **Capitolo 10** di questo libro è facoltativo. Lo leggerete se vi interessa poter controllare il Commodore 64 più di quanto non sia possibile fare con le istruzioni BASIC, o se volete sapere come funziona il calcolatore e cosa si può fare con un computer più personalizzato. Esso vi aiuterà a capire la macchina e ad aumentare le vostre capacità di avvalervene. Non è comunque essenziale per l'uso del Commodore 64.

I capitoli vertono principalmente sui comandi e le operazioni più utili per il lettore-tipo. Troverete elencati nell'**Appendice A** altri comandi disponibili per il Commodore 64 divisi per tipo e utilizzazione.

Nell'interesse della chiarezza e della facilità di comprensione si è evitato l'uso del gergo. Sarà comunque possibile che voi incontriate altri utenti di calcolatori, venditori o libri che scivolano nell'uso di termini professionali; l'**Appendice B** vi aiuterà a comprenderli.

Un decennio prima della creazione del Commodore 64 l'idea di un "personal computer" con queste possibilità era impensabile; la sua attuale versatilità dà la misura dei cabiamenti apportati dalla tecnologia. Altri cambiamenti si stanno avvicinando e, imparando a usare un calcolatore, fate un passo avanti preparandovi per questi mutamenti.

NOTA DELL'EDITORE

Per ottenere il massimo da questo libro dovete leggerlo con il vostro calcolatore davanti a voi, attrezzato e collegato a un monitor o a un apparecchio televisivo come descritto al Capitolo 1. Quando viene presentato un nuovo comando, un gruppo di comandi o un programma campione, battetelo **esattamente come stampato** se volete vedere i risultati sul vostro schermo.

Siate certi di battere tutti i caratteri che vedete: lettere, numeri, segni di interpunzione e spazi. (Nelle linee di programma composte, lo spazio equivalente a una battuta della barra spaziatrice sulla vostra tastiera varierà leggermente, ma sarà sempre approssimativamente uguale allo spazio tra le parole di questa frase). Gli spazi non sono assolutamente essenziali; i **programmi** gireranno senza di loro, ma se li userete, vi sarà molto più facile leggere sullo schermo.

Dopo che avete battuto un programma semplice, e prima d farlo girare, controllate con il libro quanto avete scritto. Se trovate qualche errore, tornate indietro e correggetelo. (Vedrete come si fa nel Capitolo 2). Tutti i programmi sono stati accuratamente controllati, e funzioneranno se immessi correttamente.

Per finire, vedrete che le parole di comando (o parole "riservate") sono state stampate in **neretto.** Scrivere in neretto è semplicemente una convenzione tipografica, tesa a richiamare la vostra attenzione su queste parole e ad aiutarvi a ricordarle. Non vedrete questo effetto sullo schermo, e quindi non avete bisogno di riprodurlo.

CAPITOLO 1

GIUNGERE AL POTERE

PADRONE E SERVITORE SI INCONTRANO

Trovandosi davanti a un calcolatore per la prima volta, le persone spesso si sentono impotenti nei confronti della macchina, e con ragione. Dopo tutto, la conoscenza è potere, e noi partiamo ignoranti.

Lavoratore incredibilmente veloce e di precisione inimmaginabile, il calcolatore è comunque un servitore assolutamente ubbidiente. Ha bisogno solo di istruzioni date in termini che sappia riconoscere. Potete comandare il Commodore 64 con un vocabolario non più ampio di quello comprensibile da un cane ben addestrato.

Un calcolatore che segue le vostre istruzioni può scegliere tra diverse alternative, organizzare informazioni, ricercare fatti ed esempi e controllare dispositivi. Gli si può ordinare di svolgere qualunque incarico in risposta a una singola parola o indicando un singolo fatto.

Nessun calcolatore è onnipossente od onniscente, e ci sono limiti all'operatività di qualsiasi macchina. Ma questi limiti derivano dalla progettazione, dalla velocità del calcolatore e dalle dimensioni della sua memoria.

Malgrado voi possiate imparare a usare il Commodore 64 in poco tempo, per farlo dovete in primo luogo imparare a diventarne padrone, ed è quello che potete fare attraverso questo libro.

INSTALLAZIONE DEL CALCOLATORE

Il Commodore 64 può essere installato da chiunque sappia sistemare una macchina da scrivere e allacciare un televisore. Può funzionare ovunque lo possa una macchina da scrivere, in ogni luogo privo di forti vibrazioni o estremi di temperatura, polvere o umidità. In effetti, il Commodore 64 funzionerà nella maggior parte dei posti dove per l'operatore è comodo lavorare.

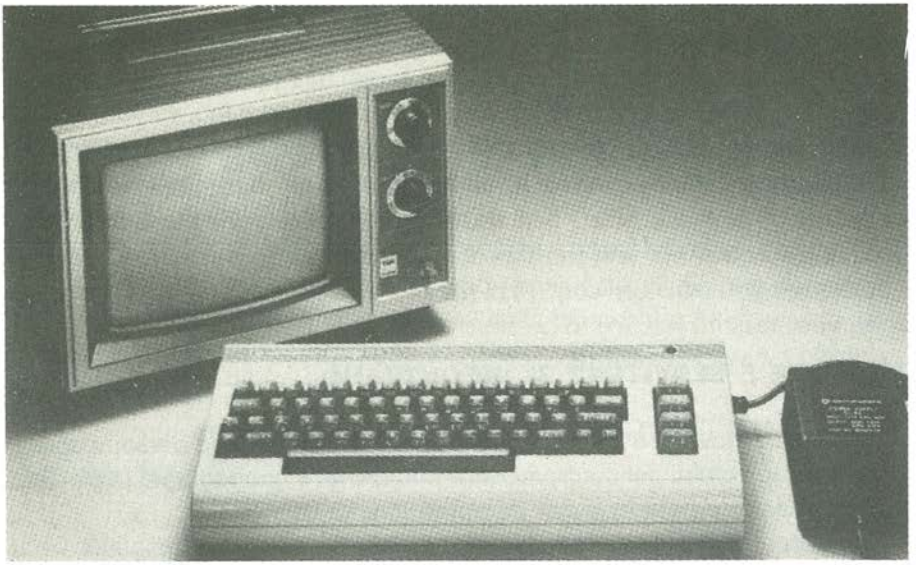

Figura 1.1 — Un sistema completo Commodore 64.

Una tipica installazione è raffigurata in figura 1.1. L'apparecchio televisivo o il monitor che fungeranno da schermo può essere situato ovunque possa arrivare il cavo del calcolatore. Potete predisporre all'uso il Commodore 64 collegando solo due cavi elettrici, uno per la corrente e uno per il televisore o il monitor che fungerà da schermo per essere situato ovunque possa stro del Commodore 64 (figura 1.2) vedrete un interruttore segnato ON, e una presa circolare vicino alla dicitura POWER. In questa presa potete inserire la spina multipla circolare del cavo di alimentazione. **Con l'interruttore in posizione OFF** potete inserire il capo a tre contatti del cavo di alimentazione nella presa di corrente a muro per completare il collegamento elettrico.

Il collegamento al televisore è altrettanto semplice. Vicino al centro del pannello posteriore del calcolatore, e vicino a un piccolo interruttore, troverete una presa metallica sporgente che ospita il cavo del televisore. Questo ha la stessa spina a manicotto metallica ad ogni estremità. Un capo va nella presa metallica del televisore sul retro del calcolatore, l'altro nella presa della scatoletta che vi permette di collegare il monitor tra la ricezione del televisore e il display del calcolatore. Da questa scatola (chiamata "switch box" scatola di scambio) escono cavi che potete collegare al televisore al posto dei cavi dell'antenna vhf. Se poi collegate i cavi vhf all'altro lato della scatola di scambio, potete scambiare tra la ricezione del televisore e il display del calcolatore. La figura 1.3 mostra il sistema collegato per il solo uso del calcolatore.

Figura 1.2 — Collegamento elettrico al calcolatore.

Figura 1.3 — Collegamenti tra calcolatore e televisore.

L'ultimo intervento include la regolazione del sintonizzatore del vostro televisore al canale 3 o 4, quello che non viene usato per la trasmissione nella vostra zona. Sistemate il canale anche sul calcolatore. Se il canale 4 non

viene usato, fate scorrere il piccolo interruttore nascosto nella parte posteriore del calcolatore (vicino alla presa metallica del televisore) verso la presa. Se invece è il canale 3, spostatelo dalla parte opposta.

Ora che il collegamento tra calcolatore e televisore è completo, potete usare il computer sistemando l'interruttore sulla scatola a COMPUTER. Per usare il televisore per ricevere le trasmissioni tv, girate l'interruttore su TELEVISION. Se usate un monitor video, un diverso cavo (terminante con una spina a cinque poli all'uscita del computer ed una spina audio all'uscita del monitor) collega a una presa circolare del video, sul retro del calcolatore vicino alla presa per il televisore, come illustrato in figura 1.4.

Una volta collegati a un apparecchio televisivo o a un monitor, il calcolatore davanti a voi è parte di un sistema operativo completo che può mostrare sullo schermo qualunque cosa voi scriviate.

CONTROLLO DELLA TASTIERA

Tutto ciò che vi occorre per comandare il calcolatore si trova davanti a voi. La tastiera è disposta come quella di una macchina da scrivere, ma ha alcuni tasti extra che vi permettono un controllo maggiore di quanto non sia possibile su una macchina da scrivere.

Prima di dare corrente al calcolatore considerate un fatto, semplice ma confortante: niente di quello che farete sulla tastiera danneggerà il computer. Con alcuni dei comandi che troverete in questo libro potrete cambiare la visualizzazione dello schermo, il contenuto della memoria attiva del calcolatore o persino il suo modo di lavorare, ma quando spegnerete il Commodore 64 e poi lo accenderete di nuovo, eliminerete tutti i cambiamenti che avevate apportato.

Il computer vi accoglie disponibile, dimentico di tutte le precedenti manipolazioni della tastiera.

Un piccolo ma utilissimo consiglio mentre imparate ad adoperare il calcolatore con la tastiera è: se vi interessa qualcosa, provate. Il peggio che vi possa capitare è la perdita di qualche istruzione, informazione o calcolo. Nei primi stadi, quando state facendo degli esperimenti, questa perdita fa parte dell'apprendimento per imparare a controllare "la bestia". Qualunque cosa accada, il calcolatore emergerà da errori umani di tastiera completamente indenne.

Quando voi accendete prima il monitor o il televisore, poi il calcolatore, apparirà sullo schermo una dicitura. Da un capo all'altro, in alto, se tutto è

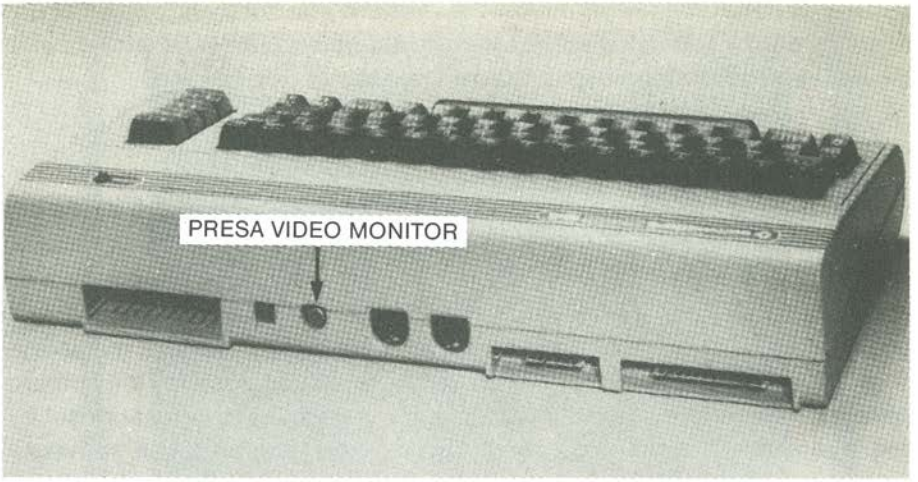

Figura 1.4 — La presa per il monitor.

collegato e funziona esattamente, verranno stampate tre righe con lettere chiare su campo scuro, come in figura 1.5.

Il cursore

Nella parte superiore sinistra dello schermo vedrete la parola READY, e sotto questa un quadrato lampeggiante delle dimensioni di un carattere. Questi sono segnali del calcolatore per indicare che è pronto a leggere qualunque cosa battiate sulla tastiera. Il quadrato lampeggiante segna dove apparirà il prossimo carattere sullo schermo ed è conosciuto come **cursore**.

Il Commodore 64 è progettato per visualizzare immediatamente quanto è stato battuto dalla tastiera e stamperà lettere azzurre su uno schermo blu così come voi le battete. La parola READY è il segnale dal calcolatore che è pronto ad accettare un comando. Ma sia che battiate uno dei segni dei comandi che può eseguire il Commodore 64, sia un riga di poesia, ciò che avete battuto comparirà immediatamente sullo schermo: e come voi battete, il calcolatore attende da voi un segnale che gli dirà di eseguire il comando che voi avete appena scritto.

Il tasto RETURN

Premendo il tasto RETURN segnalate al calcolatore di eseguire un comando. Se il comando non è più lungo di due righe, e il cursore è su una di queste linee, il calcolatore confronterà il comando battuto con una serie di i-

struzioni incorporate, e agirà di conseguenza. Ma se voi battete allegramente senza premere il tasto RETURN, il calcolatore seguirà le istruzioni che gli dicono semplicemente di visualizzare i caratteri sullo schermo.

I tasti SHIFT e C=

Allo stesso modo di una macchina da scrivere, il Commodore 64 ha sia lettere maiuscole, sia minuscole, ma a differenza di una macchina da scrivere, può avere anche dei simboli grafici. Si tratta di piccoli disegni incorniciati sulla parte anteriore della maggior parte dei tasti. Quando accendete il Commodore 64, il computer è automaticamente pronto a dare la maiuscola di ogni lettera che battete. Schiacciate il tasto con un numero, e il numero apparirà. Premete un tasto di punteggiatura, e quel segno comparirà sul video. Premendo determinate combinazioni di tasti potete anche scrivere i simboli grafici indicati sulla parte anteriore di ogni tasto. Se tenete schiacciato il tasto SHIFT contemporaneamente a un tasto con simboli grafici, comparirà il simbolo sulla destra del tasto. Tenete premuto il tasto C= (o tasto "Commodore") dopo lo SHIFT, mentre schiacciate un tasto, e comparirà il simbolo disegnato sulla sinistra.

Ogni volta che accendete il calcolatore questo è automaticamente predisposto per riconoscere battiture di tasti con le maiuscole o i segni grafici. È possibile passare a un altro sistema premendo una volta insieme i tasti SHIFT e C=, lasciandoli poi andare. Una volta fatto ciò il computer stamperà ogni battuta usando caratteri minuscoli. Premendo SHIFT e un tasto si avrà ora una lettera maiuscola, e premendo C= e un simbolo grafico si otterrà il carattere indicato sulla sinistra di quel tasto, come prima. Si esce da questo meccanismo minuscole-maiuscole per tornare al sistema maiuscole-grafici premendo ancora SHIFT e C=. In effetti, ogni volta che si schiacciano assieme i tasti SHIFT e C= voi passate da un sistema di stampa all'altro, e cambierà da un modo all'altro anche ciò che è già comparso sullo schermo.

Il tasto SHIFT LOCK può essere posto in ognuna delle due posizioni. Quando viene pigiato verso il basso e fermato in posizione abbassata ha lo stesso effetto di quando si tiene premuto il tasto SHIFT. Schiacciato ancora, il tasto SHIFT LOCK scatterà in alto e non avrà più quella funzione.

Movimenti del cursore: i tasti UP/DOWN e RIGHT/LEFT

Considerando lo schermo del Commodore 64 come l'equivalente video del foglio di carta in una macchina da scrivere, potete scrivere dappertutto entro i margini prestabiliti. Con i due tasti segnati con una freccia posti nell'angolo in basso a destra della tastiera, potete muovere il cursore, lampeggiante in

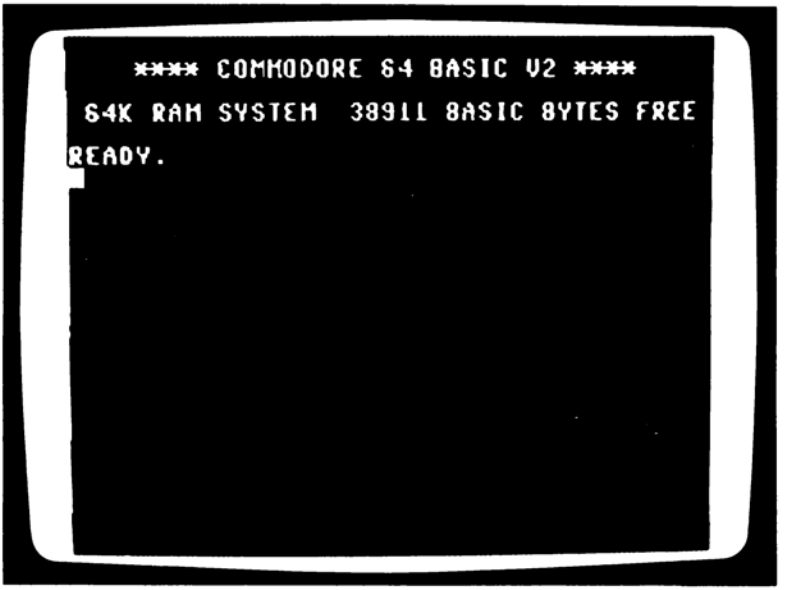

Figura 1.5 — Display che appare sullo schermo dopo l'accensione del calcolatore.

ogni punto dello schermo e qualunque cosa voi battiate comparirà in quel punto. Quando lo pigiate da solo, il tasto contrassegnato con le frecce (lo chiameremo tasto CURSOR UP/DOWN) fa muovere il cursore sullo schermo verso il basso. Allo stesso modo, il tasto contrassegnato dalle frecce destra-sinistra (che chiameremo CURSOR RIGHT/LEFT) fa muovere il cursore verso destra. Se tenuti abbassati, entrambi questi tasti continueranno a far muovere il cursore fino a quando non vengano rilasciati. Quando il tasto SHIFT, o SHIFT LOCK o C= viene tenuto abbassato mentre si preme anche il tasto CURSOR UP/DOWN il calcolatore fa muovere il cursore sullo schermo verso l'alto. Così, il tasto CURSOR RIGHT/LEFT muove il cursore verso sinistra quando viene usato in combinazione con uno qualunque di quei tre tasti. Una volta che il cursore è in fondo allo schermo, la freccia diretta verso il basso fa slittare tutte le righe scritte come fossero righe stampate su un rotolo di carta senza fine alimentato da una macchina da scrivere.

Inserire, cancellare e azzerare: INST/DEL e CLR/HOME

Altri due tasti vi consentono di controllare ulteriormente lo schermo. Il tasto posto nella parte superiore destra della tastiera indicante INST DEL ha due funzioni, può infatti inserire o cancellare. Quando viene schiacciato da

solo, indica al cursore di cancellare e avvicinare di uno spazio alla sinistra del cursore. Se viene invece premuto contemporaneamente al tasto SHIFT, questo ordina al calcolatore di aprire uno spazio alla sinistra del cursore che si sposta lì, in modo che possa venir stampato un altro carattere. Se tenete premuto questo tasto, esso ripeterà la sua azione fino a quando non lo farete scattare.

Anche il tasto contrassegnato CLR/HOME funziona in due direzioni. Potete usarlo per ordinare al calcolatore di mandare il cursore "a casa" nell'angolo in alto a sinistra oppure, tenendolo pigiato assieme al tasto SHIFT (SHIFT - CLR/HOME) dare istruzioni al calcolatore di pulire lo schermo e mandare il cursore su nell'angolo.

Altri tasti di controllo

I tre tasti segnati RUN STOP, CONTROL e RESTORE possono essere usati anche per modificare le azioni del calcolatore quando è sotto il controllo di un programma.

I quattro tasti all'estrema destra e contrassegnati f1 f2, f3 f4, f5 f6 ed f7 f8 possono essere singolarmente chiamati dai programmi per un controllo del calcolatore in modo appositamente programmato.

Ancora a proposito del tasto RETURN

Con un segnale al calcolatore di operare su quanto avete battuto alla tastiera, il tasto RETURN agisce da messaggero avviando le istruzioni della tastiera nell'**assemblatore** del Commodore 64. Quando premete il tasto RETURN dopo aver scritto qualcosa in risposta a una richiesta di READY, l'assemblatore dell'elaboratore confronta quanto avete inviato con il glossario che ha incorporato. Se la forma della vostra istruzione corrisponde a un comando, l'assemblatore passa gli ordini alla parte elaboratrice del computer, che agisce in conformità alle istruzioni incorporate.

Potete far muovere il cursore dappertutto sullo schermo e scrivere qualunque cosa vi piaccia. Se schiacciate il tasto RETURN e l'assemblatore non esegue le vostre istruzioni con il suo vocabolario, il calcolatore vi segnalerà che c'è un disguido. Nel Commodore 64 questo segnale è "?SYNTAX ERROR". Questo tipo di risposta significa che si è verificato un errore di comunicazione. ("Syntax" si riferisce all'esatta forma o sequenza nella quale vanno date le istruzioni). In ogni caso, troverete sempre utile il tasto RETURN ogni volta che sarete pronti ad impartire istruzioni al calcolatore.

Figura 1.6 — La tastiera del Commodore 64.

Riassunto delle operazioni alla tastiera

Ogni tasto ha una funzione (stampare un carattere o inviare un comando), ed alcuni hanno parecchi effetti diversi se usati in combinazione. È possibile ottenere programmi (serie di comandi) che cambieranno la funzione di alcuni o di tutti i tasti.

Schiacciando un qualunque tasto si invia un segnale al calcolatore che risponde stampando un carattere o compiendo una azione predeterminata.

La tastiera del Commodore 64 è illustrata in Figura 1.6. Sia che usiate comandi preregistrati (da nastri o dischi - vedi capitoli 7 e 8) o diretti, in ultima analisi effettuate sempre il vostro controllo sul Commodore 64 tramite tastiera. Essa serve da redini e briglia per il calcolatore e per ogni altra unità ad esso collegata.

CAPITOLO 2

SCRIVERE SUL VIDEO

500 anni fa Giovanni Gutemberg stabilì un modello di stampa che non è ancora gran che cambiato. In effetti, solo l'introduzione del calcolatore negli ultimi anni ha avuto un impatto maggiore. La composizione fatta col computer mette la pagina stampata sullo schermo prima che vada sulla carta. Con il vostro Commodore 64 potete creare display sullo schermo controllati da comandi.

CONTROLLO DELLO SCHERMO

Limitandosi semplicemente a riprodurre le vostre battute sullo schermo, il Commodore 64 non sfrutta nessuna delle sue capacità, ma con comandi adeguati, tuttavia, potete guidare il calcolatore a visualizzazioni pre-programmate. Il computer reagisce istantaneamente ad alcuni comandi inviati premendo il tasto RETURN. Tra questi comandi "immediati" ci sono quelli che vi permettono di modellare ciò che comparirà sullo schermo.

Potete riempire il video con qualunque carattere battendo sulla tastiera senza mandare al calcolatore alcun comando. Se stampate più di quanto lo schermo possa contenere, il computer farà muovere all'insù l'intero testo di una riga per far posto all'ultima, e la prima in alto andrà perduta.

Potete comandare il Commodore 64 con singoli comandi di battuta o di parole. Premendo i tasti SHIFT e CLR/HOME contemporaneamente, per esempio, ordinerete al calcolatore di ripulire lo schermo da tutto ciò che è stato scritto e di riportare il cursore lampeggiante nell'angolo in alto a sinistra. Dopo questo comando, il computer presenterà uno schermo vuoto, qualunque cosa fosse stata scritta prima. Se per esempio voi foste poeti e scriveste qualcosa come i versi in figura 2.1, potreste cancellare subito quando avevate scritto facendo così:

(premere SHIFT e CLR/HOME)

Comandato da queste istruzioni, il calcolatore cancella tutto, lasciando uno schermo vuoto con il cursore nell'angolo superiore sinistro — come se fosse un foglio di carta bianco. Quando battete caratteri di tastiera, il computer li invia allo schermo sotto forma di punti colorati chiari su uno sfondo scuro e vi risponde allo stesso modo, automaticamente, con lettere chiare su base scura.

Esistono tuttavia altri modi di presentare un testo sullo schermo, utili per variare il risalto dei caratteri e richiamare l'attenzione su alcune parti del testo.

Se volete far apparire caratteri scuri su uno sfondo chiaro, dovete eseguire i seguenti comandi:

(premere CTRL e 9)

e tutto ciò che segue verrà stampato sullo schermo in modo "invertito". Il comando CTRL-9 controlla il calcolatore come un interruttore così che, fino a quando non verrà inviato un altro comando a cambiare l'impostazione (o fino a che non viene premuto il tasto RETURN), tutto ciò che verrà scritto appairà sullo schermo in caratteri scuro-su-chiaro. I caratteri che erano stati battuti prima di quel comando saranno chiaro-su-scuro, che è poi il modo normale di scrivere a meno di istruzioni diverse. Il poeta Robert W. Service, impartendo il comando CTRL-9 all'inizio di un verso, avrebbe visto uno schermo uguale a quello in figura 2.2.

Potete riportare il calcolatore al modo normale chiaro-su-scuro con il seguente comando:

(premete CTRL e 0)

che ordinerà al computer di stampare chiaro-su-scuro tutti quei caratteri che verranno scritti dopo. Lo stesso poeta, continuando la composizione del suo poema dopo aver premuto CTRL e 0, vedrebbe lo schermo come in figura 2.3. Potete anche creare testi multicolori. Usando i tasti numerati da 1 a 8 assieme al tasto CTRL, è possibile scegliere colori differenti per i caratteri che compaiono sullo schermo. Per avere lettere gialle, per esempio, dovete premere contemporaneamente i tasti CTRL e 8:

(premere CTRL e 8)

Se il calcolatore è nel modo chiaro-su-scuro, tutti i caratteri stampati dopo il comando CTRL-8 appariranno come lettere gialle su uno sfondo blu scuro. Se viene dato l'ordine di invertire il display, CTRL-9, si vedranno lettere blu scuro, ognuna nel suo blocco giallo. Potete mutare i caratteri in altri colori

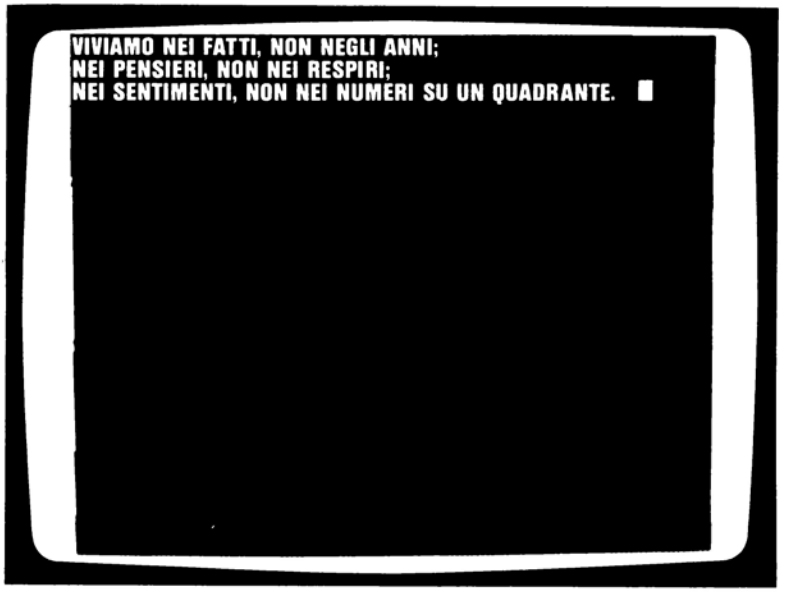

Figura 2.1 — Un esempio di display.

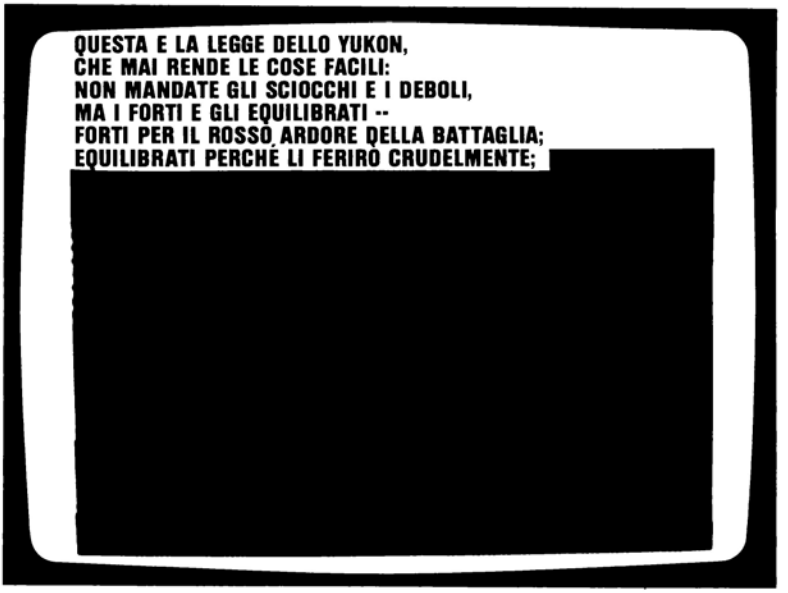

Figura 2.2 — L'effetto del comando CTRL-9.

premendo il tasto CTRL unitamente a qualunque altro tasto numerato. Schiacciando il tasto C= e uno dei tasti numerati, il calcolatore potrà fornire anche altri colori.

STAMPARE SUL VIDEO

Il Commodore 64, come ogni altro calcolatore degno dei suoi circuiti, può fare di più che visualizzare le vostre battiture di tasti sullo schermo. Infatti la maggior parte del suo potere di calcolo sta nelle operazioni incorporate, sulle quali lavora furiosamente, a comando, senza che tutta questa fatica sia visibile. L'esempio più semplice si può vedere nell'aritmetica elementare, su cui il Commodore 64 lavora automaticamente, e non visto, quando ne riceve l'istruzione. Le solite funzioni di addizione, sottrazione, moltiplicazione e divisione sono comandate dalla tastiera mediante i simboli +, –, *, e /. Se voi dite al calcolatore di moltiplicare –2 e 4, scrivendo

–2 * 4

non vedrete il risultato, ma una risposta ?SYNTAX ERROR e un'altra richiesta READY. Per vedere il risultato di quella moltiplicazione è necessario aggiungere un comando.

Il comando PRINT

Il comando che ordina al calcolatore di mostrare il risultato dell'operazione aritmetica sullo schermo è PRINT. Può essere usato con un comando dell'"operatore", come il simbolo della moltiplicazione *, per mostrare una delle operazioni interne del calcolatore. Bisogna battere questo utilissimo comando come prima cosa, prima dell'operazione che volete sia effettuata, così:

PRINT –2 * 4

Il comando PRINT ordina al calcolatore di mostrare sullo schermo il risultato della moltiplicazione che segue. Dopo aver mandato quest'ordine premendo il tasto RETURN, vedrete comparire sullo schermo il valore –8 e una richiesta READY sotto di esso, indicante che il calcolatore è pronto per un altro comando, come illustrato in figura 2.4.

Potete usare il comando PRINT per qualsiasi combinazione di comandi aritmetici secondo le regole dell'algebra utilizzando il vostro sofisticato Com-

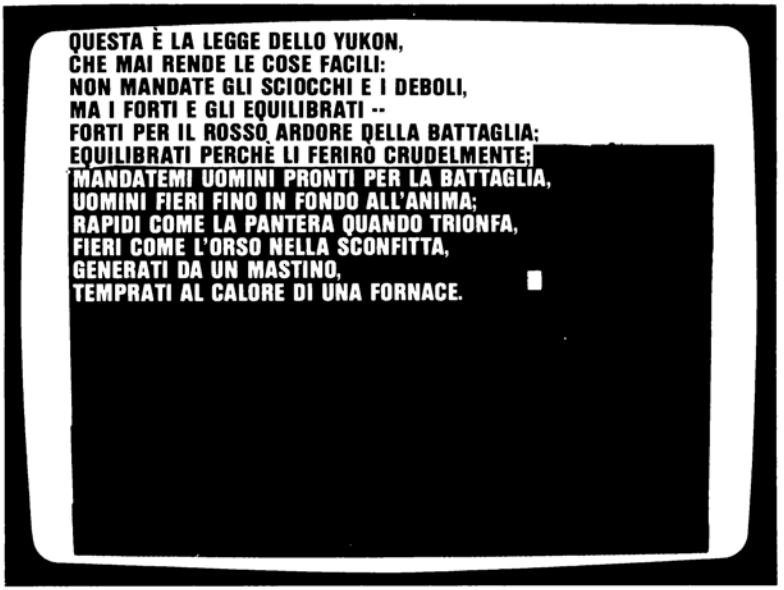

Figura 2.3 — L'effetto del comando CTRL-0.

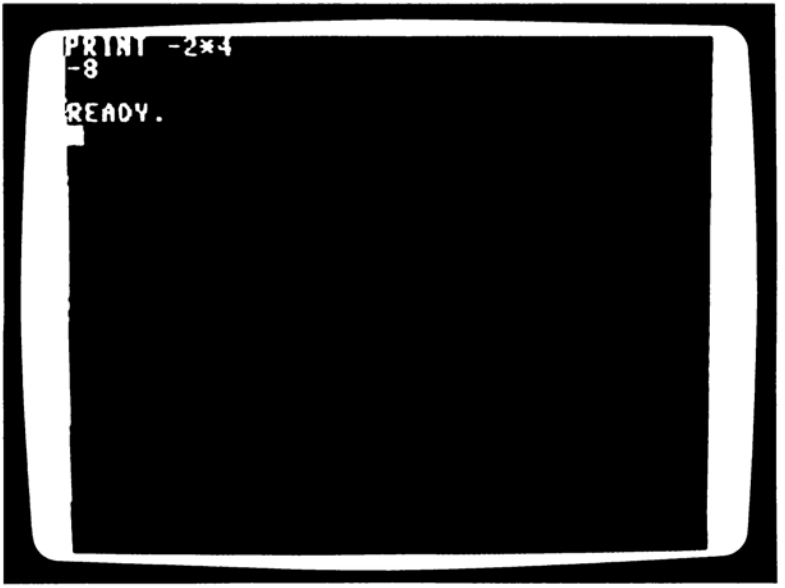

Figura 2.4 — L'effetto del comando PRINT.

modore 64 come una semplice macchina calcolatrice con immissioni come questa:

PRINT 9 + 8 - 7*6/5

Il calcolatore segue automaticamente un ordine di operazione predeterminato nell'interpretazione di espressioni aritmetiche come questa. Vengono svolte prima la moltiplicazione, poi la divisione ed infine addizione e sottrazione per avere in questo caso sullo schermo come risultato 8.6. Per modificare questa sequenza incorporata di operazioni è necessario l'uso delle parentesi, per ordinare al computer di elaborare per prima cosa i valori in esse contenuti, così:

PRINT (9+8–7)*6/5

e ottenere il risultato che comparirà, sotto l'istruzione PRINT, di 12.

Ma l'utilità del comando PRINT va ben al di là della semplice esibizione di manipolazioni matematiche (tutte elencate nell'Appendice A). PRINT è un comando che potete usare spesso. Oltre ad usarlo per effettuare alcune operazioni sullo schermo, potete adoperarlo per esprimere un sentimento, un pensiero o un fatto nel bel mezzo di qualche altra azione del calcolatore. Il comando PRINT è il vostro dattilografo.

Potete ordinare al calcolatore di scrivere sullo schermo qualunque cosa vogliate **tra virgolette** dopo il comando PRINT. Per esempio

PRINT "Sono lieto di essere qui"

La risposta del calcolatore dopo aver premuto il tasto RETURN sarà,

Sono lieto di essere qui

sulla riga seguente il comando.

In una istruzione PRINT si possono combinare assieme parole e numeri. Per esempio, potete dare l'istruzione:

PRINT "Il numero vincente è ....." 30

e vedere il calcolatore rispondere:

Il numero vincente è ..... 30.

Potete anche infilare un calcolo in una frase stampata, così:

PRINT 20 "o" 30" è più di quanto intenda pagare. Ma "10+5" andrebbe bene".

e vedrete comparire sul computer:

20 o 30 è più di quanto intenda pagare. Ma 15 andrebbe bene.

Comandi di tasto nell'istruzione PRINT

Potete caricare un'istruzione PRINT con comandi di tastiera racchiudendoli tra virgolette e il calcolatore non li eseguirà fino a che non avrete premuto il tasto RETURN per mandare l'intero comando PRINT.

Perciò un'istruzione PRINT deve tenere in sospeso delle istruzioni per fare diverse cose. Per esempio, potete comandare al calcolatore di ripulire lo schermo, commutare il display a caratteri scuro-su-chiaro, poi passare al giallo, quindi stampare una frase, girare sul bianco, poi stampare un'altra frase, tutto con un comando come questo:

PRINT "(premetre SHIFT-HOME) (premere CTRL e 9) (premere CTRL e 8) ↑↑↑ PRENDILO (premere CURSOR UP/DOWN) (premere CURSOR UP/DOWN) (premere CTRL e 2) DALL'ALTO"

TAB - Sistemare il video-schermo

Se inviate un comando attraverso la tastiera, come SHIFT-CLR/HOME tra virgolette in un'istruzione PRINT, il calcolatore registrerà questo comando in maniera codificata, in questo caso con un simbolo a forma di cuore, ma non lo eseguirà finché non verrà premuto il tasto RETURN. Potete usarlo per ritardare la "confezione" di un display con il comando PRINT. Una volta che vi sarete resi conto che il calcolatore tratta il testo sullo schermo come un reticolo di 40 colonne di 25 righe, potrete ideare delle disposizioni che appaghino anche l'occhio.

Un comando, piccolo ma utile, usato in un'istruzione PRINT, ordina al calcolatore di **tab**ulare prima di scrivere sullo schermo. La forma è TAB ( ) dove le colonne sono numerate, come in una macchina da scrivere, da sinistra a destra, da 0 a 39. Potete aggiungere il comando TAB prima dell'elemento da stampare, così:

PRINT "HEY!" **TAB** (15) "LOOK" **TAB** (30) "AT THIS"←

Il calcolatore agisce su questa istruzione leggendola attraverso una serie di comandi, e stampa:

HEY! LOOK AT THIS←

Se un comando è già comparso sullo schermo, potete ripeterlo muovendo il cursore (con il tasto CURSOR UP/DOWN) alla riga dello schermo dove si

trova già il comando. Se poi premete il tasto RETURN, inviate di nuovo al calcolatore l'istruzione di agire. Potete persino alterare un comando già apparso sullo schermo, e quindi inviarlo premendo il tasto RETURN. Potreste, per esempio, far muovere il cursore su per lo schermo fino al comando PRINT di cui sopra usando i tasti SHIFT e CURSOR UP/DOWN. Utilizzando il tasto CURSOR RIGHT/LEFT per muovere il cursore orizzontalmente e SHIFT e INST/DEL per inserire spazi, potete cambiare l'istruzione. Un modo di cambiare l'istruzione dovrebbe essere tale che il display dovrebbe risultare:

HEY! TAKE A LOOK AT THIS←

Per fare ciò inserite la parola TAKE A nel secondo gruppo di virgolette. Potete far ciò ponendo il cursore sulla lettera L della parola LOOK e inserendo sette spazi per le parole TAKE A e lo spazio tra loro. Per inserire gli spazi, premete semplicemente i tasti SHIFT e INST/DEL assieme per sette volte:

(premere SHIFT e INST/DEL) (premere SHIFT e INST/DEL)
(premere SHIFT e INST/DEL) (premere SHIFT e INST/DEL)
(premere SHIFT e INST/DEL) (premere SHIFT e INST/DEL)
(premere SHIFT e INST/DEL)

Poi potete effettuare l'inserimento

TAKE (premere SPACE BAR) A

Il comando che rimane dopo questi cambiamenti è:

PRINT "HEY!" **TAB** (15) "TAKE A LOOK" **TAB** (30) "AT THIS←"

Se ora schiacciate il tasto RETURN vedrete apparire un display modificato:

HEY! TAKE A LOOK AT THIS←

Potete usare questo metodo per inviare qualunque comando dallo schermo al calcolatore. Se l'istruzione di comando occupa fino a due righe sullo schermo, potete mandarla posizionando il cursore su qualunque punto di una delle due righe, premendo poi il tasto RETURN.

Altri comandi

Due segni di interpunzione possono essere usati come comandi abbreviati all'interno di un'istruzione PRINT. Virgole scritte tra articoli in un comando

PRINT ordineranno al calcolatore di spaziare gli articoli sullo schermo come se si seguisse una successione di comandi TAB (2), TAB (12), TAB (22) e TAB (32). Punti e virgole tra articoli li terranno separati nell'istruzione, ma non nella stampa. Inoltre, il calcolatore aggiunge automaticamente uno spazio prima e dopo i numeri che stampa. Il calcolatore e il suo schermo forniscono un display più facilmente controllabile di quanto non sia la carta nella macchina da scrivere. Potete comandare i colori che dovranno comparire, e perfino la loro rapida successione, fissando comandi di cancellazione dello schermo tra un display e un altro.

Usando ulteriori comandi potete guidare il calcolatore in qualsiasi direzione. Con l'uso di comandi in combinazione, potete perfino fargli eseguire i vostri ordini da solo.

CAPITOLO 3

PIANI E PROGRAMMI

COMANDI DI SCHEDULAZIONE

Il calcolatore, per lungo tempo considerato un terrificante accumulatore di informazioni e segreto custode di archivi, è fondamentalmente smemorato. Per meglio dire, è costruito per agire su comandi, che poi si lascia automaticamente alle spalle per essere disponibile per altri ordini. Anche molti esseri umano lavorano allo stesso modo, ritornando in posti ricordati solo vagamente grazie a un indirizzo che è rimasto in mente.

A modo suo, il calcolatore può tornare a indirizzi rimasti nella sua memoria. Se trova ordini ad attenderlo a questi indirizzi, esso agirà su di loro come se li aveste appena mandati. Potete procurarvi tali indirizzi facendo iniziare ogni gruppo di istruzioni con un numero.

Posto all'inizio di un gruppo di comandi, un numero dice al calcolatore di considerare questi comandi come facenti parte di un **programma** che potrà venire usato ancora. Per esempio, per indicare un'istruzione in questo modo, potreste mandare il seguente comando:

```
10 PRINT "(premere SHIFT e CLR/HOME)" TAB (12)
   "NUMERI E LORO UTILIZZI"
```

Premendo il tasto RETURN dopo aver scritto un'istruzione come questa, noterete che il calcolatore risponde con il cursore intermittente, senza agire sui comandi. In realtà, l'istruzione viene immagazzinata nella memoria, dove è tenuta in archivio per numero. Qui, gli ordini saranno conservati come riga numero 10 finchè non scriverete il comando che dirà al Commodore 64 di procedere. Questa istruzione è

```
RUN
```

che ordina al calcolatore di tradurre e agire sulle istruzioni numerate nella memoria in ordine numerico. Quale unica riga numerata degli ordini in me-

moria, in questo caso, la riga 10 verrà azionata comando per comando. Il calcolatore tiene automaticamente in memoria gli ordini numerati, anche dopo aver agito su di essi, di modo che è possibile dire al computer di eseguire le istruzioni di riga 10 tutte le volte che volete, mandando ogni volta il comando RUN.

Anche se le istruzioni rimangono invisibili nella memoria del calcolatore, potete evidenziarle sullo schermo con un semplice ordine. Se mandate l'istruzione

LIST

il calcolatore vi risponderà stampando l'intero contenuto della sua memoria di programma, in questo caso la riga 10. Potete mandare un'altra serie di comandi per far agire il computer oltre la riga 10 contrassegnando l'istruzione con un numero più alto, così:

20 **PRINT** "(premere CTRL e 9) (premere CURSOR UP/DOWN) "QUESTA RIGA VERRÀ ELIMINATA"

Il calcolatore ha ora due righe in memoria, e quando riceverà il comando

RUN

comincerà dall'istruzione con il numero più basso, la 10, tradurrà ed effettuerà ogni comando, e poi tradurrà ed effettuerà i comandi dell'istruzione 20.

Potere governare il calcolatcre con una serie di istruzioni numerate commutando tra diversi display sullo schermo per ottenere un risultato simile a quello prodotto da queste istruzioni supplementari:

30 **PRINT** "(premere CTRL e 8) "PER VOSTRA INFORMAZIONE, ORA STATE PROGRAMMANDO".
40 **PRINT** "IL PROGRAMMA (premere C= e 7) FINISCE QUI"

Se, dopo aver mandato queste istruzioni al calcolatore, date il comando RUN, vedrete una scritta di quattro istruzioni apparire una dopo l'altra con il risultato finale illustrato in figura 3.1.

Le righe di istruzione 10, 20, 30 e 40 formano un programma che può dare ordini al calcolatore. Otterrete lo stesso display ogni volta che emetterete il comando RUN. Cambiate le istruzioni del programma, e ne modificherete il risultato. Per esempio, la riga scuro-su-chiaro che dice:

QUESTA RIGA VERRÀ ELIMINATA.

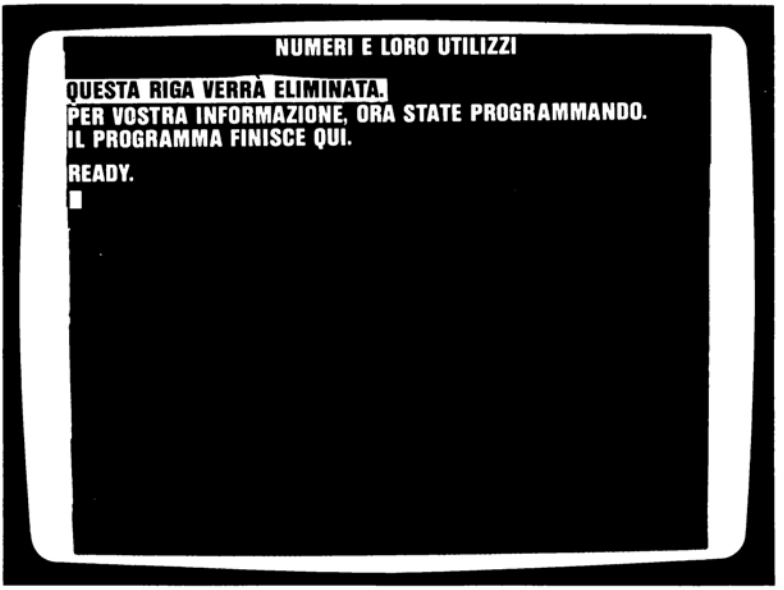

Figura 3.1 — Risultato dello svolgimento di un programma di comandi che agiscono sul display.

non comparirà sullo schermo durante lo svolgimento del programma se cancellate riga 20, in cui avete ordinato la sua stampa. Per cancellare una riga dalla memoria di programma del calcolatore potete semplicemente scrivere il numero di riga e premere il tasto RETURN:

20

Il calcolatore resta in attesa di accettare una nuova istruzione e quando fate ciò si prepara a immagazzinare una sequenza di ordini al di sotto del numero 20.

Quando mandate solo il numero, esso sostituisce la prima istruzione avente questo numero con un'istruzione nuova, vuota. Poichè un'istruzione vuota non dà ordini, la riga 20 cessa di esistere nella memoria del programma.

Quando farete girare (RUN) il programma, vedrete che il testo scuro-su-chiaro di riga 20 mancherà e quando stamperete un elenco istruzioni (LIST), troverete in memoria solo le rimanenti righe 10, 30 e 40, come illustrato in figura 3.2.

Il calcolatore segue questa semplice regola nell'utilizzare le righe di programma: il contenuto di una nuova riga numerata sostituisce il contenuto precedente dello stesso numero. Potete aggiungere una nuova istruzione a riga 20 così:

20 **PRINT** "111"

per includere le tre frecce nel display dopo che è stata stampata la riga:

NUMERI E LORO UTILIZZI.

Il calcolatore svolge le istruzioni numerate in modo sequenziale, seguendo qualunque schema prepariate (di positivi, di numeri interi fino a 63,999). Seguirà un programma iniziando da qualunque numero risalendo la sequenza, senza considerare gli eventuali numeri mancanti. La numerazione per dieci è un facile approccio alla programmazione che lascia spazio tra le istruzioni per aggiungere quelle righe di programma che possono servirvi in un ripensamento o un perfezionamento. Il calcolatore tiene le righe di istruzione numerate nella sua memoria di programma finchè non togliete l'alimentazione, o levate alcune righe stampando il loro numero di riga, od ordinate al calcolatore di cancellarle tutte.

Potete richiedere questa cancellazione inviando semplicemente il comando:

NEW

Quando riceve questo comando il computer si prepara per eseguire un nuovo programma cancellando tutte le righe dalla sua memoria. Se scrivete un nuovo programma senza cancellare il vecchio, i numeri di riga del vecchio, non sostituiti dai nuovi, si aggiungeranno al programma originale dando istruzioni che potreste non aver progettato.

SOSTITUTI DEI NUMERI

I numeri, come avrete senza dubbio notato, sono utili sia all'interno di un comando, sia per individuare le righe di programma e, come i comandi, i valori numerici aiutano a controllare le operazioni svolte dal calcolatore. Il cambiamento del valore di un numero modifica l'azione del Commodore 64, per esempio, nelle istruzioni date da un comando TAB o nella moltiplicazione richiesta con un comando ad asterisco (*).

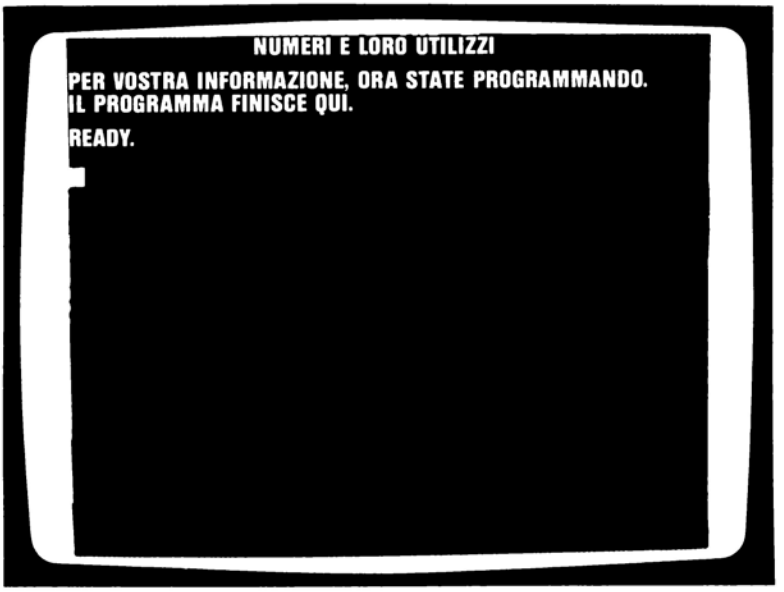

Figura 3.2 — Una lista di programma dopo che è stata eliminata una riga.

Un singolo numero può passare attraverso un programma, per essere usato in comandi diversi, se adoperate un simbolo che sta al posto del numero che vorreste far agire. Se voi informate il calcolatore all'inizio del programma che un simbolo rappresenta un numero, il computer sostituirà il numero ad ogni comando in cui in seguito incontrerà quel simbolo, e poi lo elaborerà.

Il simbolo più comodo da usare al posto di un numero è una lettera, che potete usare in tutti i comandi dove avreste altrimenti adoperato il numero.

Per esempio, se state esaminando la posizione di un testo sullo schermo, potete porre tale sostituto in un comando TAB all'interno di un programma, così:

```
20 PRINT "(premere SHIFT e CLR/HOME)"
30 PRINT TAB(A) "COME SEMBRA?"
```

Qui la lettera A sostituisce un numero nel comando TAB. Se lanciate l'istruzione RUN per questo programma di due righe senza aver assegnato un valore ad A, il calcolatore seguirà le istruzioni incorporate e sostituirà la lettera A con un valore zero.

L'istruzione di assegnazione

Il comando mediante il quale viene dato un valore a un simbolo è conosciuto come "istruzione di assegnazione". È rappresentato da un segno di uguale (=) e, per ordinare al calcolatore di dare al numero 12 il valore di A in questo programma, sarà necessario mandare il seguente comando:

```
10 A = 12
```

Quando fate girare questo programma di tre righe, il risultato sarà dato dalle parole "COME SEMBRA?" stampate a partire dalla colonna 12.

Avendo chiamato così un numero usato nel programma, potete cambiare il suo sistema operativo senza modificare la riga che ordina questa operazione. Cambiate il valore di A in riga 10, così:

```
10 A = 20
```

e la macchina stamperà alla colonna 20. Usata in questo modo, la lettera A agisce come un simbolo **variabile** che può indicare qualsiasi numero voi specifichiate con un identico segno.

Potete usare una variabile in ogni comando del vostro programma. Se al programma aggiungete il comando:

```
25 PRINT "COLONNA A"
```

il calcolatore risponderà, quando farete girare il programma, scrivendo:

```
COLONNA 20
```

allora stamperà la domanda che inizia in colonna 20 sotto quella riga. Ancora, se cambiate il valore di A e date il comando RUN, il calcolatore posizionerà il testo di conseguenza, per numero di colonna:

```
10 A = 5
```

Una variabile come A può essere usata in un programma una quantità infinita di volte come sostituta di un numero. Qui potete usarla per provare diverse posizioni della stampa sullo schermo riscrivendo solo il piccolo programma di riga 10. In programmi più complessi, cambiare il valore di una variabile modificando un comando di assegnazione può avere un effetto profondo.

Il comando di assegnazione può essere tradotto come un'istruzione al calcolatore di lavorare all'interno per sostituire il simbolo sulla sinistra del segno di uguale (ovunque si incontri questo simbolo) con il valore numerico sulla destra del segno stesso.

PROIEZIONI NUMERICHE

Il calcolatore può agire su più di una variabile simbolica se includerete un'istruzione di assegnazione per ciascuna. Ciò vi dà l'enorme potere di scrivere programmi per analizzare o prevedere situazioni diverse. Se è possibile descrivere una situazione con i numeri, è anche possibile ordinare al calcolatore di simularla modificandone uno o più aspetti. Potete, per esempio, confrontare i risultati di differenti offerte di stipendio o il riempimento di serbatoi di varie capacità.

La previsione del vostro conto in banca può essere fatta creando un modello numerico dei fattori noti da elaborare. Poi, inviando al calcolatore i valori di questi fattori mediante istruzioni di assegnazione, vedrete quale risultato il modello prevede. Potete facilmente prevedere la quantità di acqua in un serbatoio o l'ammontare del vostro conto, diciamo dopo nove mesi, se conoscete la cifra attualmente esistente e la quota media mensile di incremento. Con il seguente programma potete ordinare al calcolatore di stampare questi valori e poi scrivere il calcolo dell'ammontare totale nove mesi dopo. (Assicuratevi di inviare il comando NEW per cancellare dalla memoria il programma precedente prima di scrivere quello nuovo). Qui, la cifra di partenza sarà rappresentata dalla lettera A e la quota mensile di incremento con la lettera B:

```
NEW
30 PRINT "(premere SHIFT e CLR/HOME)"
40 PRINT "PARTENDO CON" A
50 PRINT "E GUADAGNANDO" B "OGNI MESE"
60 PRINT "AVRETE" A + (B * 9) "DOPO 9 MESI."
```

Se ora fate girare il programma, esso sostituirà automaticamente zero alle variabili A e B. Ma se assegnerete valori alla cifra di partenza e alla quota di incremento, il calcolatore evidenzierà questi valori e, attraverso le istruzioni di riga 60, scriverà una proiezione dell'ammontare futuro. Potete assegnare valori alle variabili mediante le righe:

```
10 A = 2200
20 B =  355
```

Il calcolatore farà girare l'intero programma considerando 2200 come importo di partenza (galloni d'acqua, o dollari) e 355 come incremento mensile per produrre il display illustrato in figura 3.3.

Potete, ovviamente, cambiare la simulazione modificando i valori assegnati alle righe 10 e 20, e fare quindi proiezioni di situazioni diverse. Questo programma è stato scritto per produrre una proiezione su nove mesi, ma può essere cambiato sostituendo il valore 9 con una variabile che rappresenti un numero diverso di mesi. Per fare ciò è necessario modificare riga 60 che produce il display e i calcoli. È possibile assegnare la variabile C al numero di mesi:

```
60 PRINT "AVRETE" A + (B * C) "DOPO" C "MESI."
```

Aggiungendo un altro ordine di assegnazione potete specificare un altro periodo di tempo, diciamo 15 mesi:

```
25 C = 15
```

Ora, quando farete girare il programma, questo mostrerà l'importo di partenza, 2200, e prevederà un aumento di 355 al mese, su un periodo di 15 mesi.

Potete ordinare al computer di sostituire una variabili con un'altra, o con un'espressione, che consiste in variabile e operatori. Se in questo programma voleste un modo più breve per indicare l'ammontare finale previsto, dovreste ribattezzare questo importo rappresentato dal valore di A + (B × C), con un'istruzione separata, chiamata F:

```
27 F = A + (B * C)
```

ora che il valore finale ha un nome semplice, potete ordinare sia il calcolo, sia il display della differenza tra i valori originale e finale, A ed F, con questa semplice istruzione:

```
70 PRINT "IL GUADAGNO E'" F - A
```

Potreste poi delineare la forma del programma mediante una sostituzione in riga 60:

```
60 PRINT "AVRETE" F "DOPO" C "MESI."
```

Figura 3.3 — Risultati di un programma di previsione.

È possibile cambiare ancora i dettagli modificando i valori delle righe 10, 20 e 25.

Anche se finora abbiamo usato solo numeri interi, il calcolatore è stato progettato per lavorare anche su cifre decimali. Può considerare valori molto dettagliati, per esempio A = 2200,35; può anche lavorare su frazioni se queste vengono trattate come divisioni. Per esempio, il valore di mesi 10 3/4 sarà detto C = 10 + 3/4.

COMANDI RAGGRUPPATI

I comandi funzionano perfettamente da uno a una riga numerata di programma, ma potete anche combinarne diverse assieme per convenienza o per comodità di programmazione.

Le tre istruzioni di assegnazione del programma precedente:

```
10 A = 2200
20 B =  355
25 C =   15
```

se lo preferite possono venire raggruppate in un singolo numero di riga.

Se i comandi sono tenuti separati da due punti, il calcolatore agirà su ogni singolo comando alla volta, come se ognuno di loro fosse su una propria riga numerata. È possibile sostituire queste tre righe con una sola, così:

```
10 A = 2200 : B = 355 : C = 15
```

Bisognerà poi cancellare le due righe ora superflue inviando numeri di righe vuote mediante il tasto RETURN:

```
20
25
```

Il nuovo numero di riga 10, che comprende un gruppo di comandi, sostituisce ora le precedenti righe 10, 20 e 25. Il calcolatore conterrà nella memoria le stesse informazioni ed eseguirà i comandi del programma esattamente allo stesso modo. Potete raggruppare in un unico programma tanti comandi quanti ne potrà contenere lo spazio di 80 caratteri che il calcolatore riconosce per le note di comando.

Anche se entrambi i metodi portano allo stesso risultato, potete preferire un sistema all'altro. In un programma scritto con un comando per riga è più agevole trovare i problemi che richiedono ritrascrizione o modifica, come per esempio l'accidentale errore che porta a una risposta di "?SYNTAX ERROR". D'altra parte, raggruppare diversi comandi su una riga può essere un modo più semplice di comporre un programma per istruzioni a comando multiplo. In entrambi i casi, la scelta sta a voi.

CAPITOLO 4

CAPACITÀ DI DECISIONE DEL CALCOLATORE

USARE DISCERNIMENTO

Potete controllare il vostro calcolatore al pari di una sentinella, decidendo quando volete che alcuni comandi vengano ripetuti, o i valori cambiati, o quando volete interrompere il corso di un programma. Ma questo significa che voi lo state assistendo, piuttosto che farvi assistere da lui. Per fortuna, questa vigilanza non è necessaria. Potete riflettere le vostre decisioni in un programma, così che quando il calcolatore incontra quelle particolari condizioni che avete descritto, questi prenderà le decisioni che avete indicato. Il Commodore 64 può esaminare un confronto numerico (l'affermazione cioè che un valore è uguale o disuguale, maggiore o minore di un altro) e "valutarlo" come vero o falso. Se l'affermazione è vera, verrà eseguita l'istruzione che immediatamente segue. In caso contrario, il calcolatore passerà alla riga numerata seguente.

Il comando che consente questa capacità di decisione è l'istruzione IF-THEN che ha la stessa forma dell' "if ... then" della lingua inglese: "If it's raining now, then go by car." **Se** l'affermazione è vera, **allora** prendi la decisione indicata, in caso contrario, no. Il comando del calcolatore appare in questa forma:

IF (confronto) **THEN** (istruzione)

Vedremo via via degli esempi specifici.

Potete usare il comando IF-THEN in un nuovo programma per valutare una somma di tre numeri rappresentata come segue:

```
10 A = 5 : B = 8 : C = 87
20 T = A + B + C
```

Potete poi predisporre le cose in modo che il calcolatore indichi una istruzione se il totale è 100, con il comando:

```
30 IF T = 100 THEN PRINT "UN TOTALE DI CENTO"
```

che istruirà il calcolatore a stampare la proposizione solo se si verifica la condizione T = 100.

Quando questo programma sarà stato fatto girare nel Commodore 64, si vedrà stampata la parte racchiusa tra virgolette. Se la riga 10 viene cambiata, così che i valori vadano oltre la cifra stabilita, il calcolatore salterà THEN e il suo comando PRINT, arrivando in questo caso alla fine del programma.

Incontrando un'istruzione IF-THEN il calcolatore segue delle istruzioni incorporate per valutare la condizione affermata alla destra della parola IF. Quando questa condizione è vera, il calcolatore segue il comando alla destra della parola THEN. Quando invece è falsa, il computer salta alla riga seguente.

Potete indirizzare il calcolatore in questo processo decisionale attraverso relazioni matematiche per la cui comprensione è stato costruito. La seguente modifica del programma, per esempio, analizza il numero di ospiti in arrivo a gruppi per un raduno:

```
30 IF T > 100 THEN PRINT "LA STANZA STA DIVENTANDO
   AFFOLLATA"
```

Qui, l'analisi non è se T è **uguale a** un valore particolare, ma se è **maggiore di** un dato valore.

Quando il calcolatore incontra un numero dopo la parola THEN, esso lo traduce come un'istruzione di proseguire con il programma partendo dai comandi di questo numero di riga. Così, oltre a seguire immediatamente un comando dell'istruzione IF-THEN, il calcolatore può essere inviato a un'altra parte del programma per numero di riga, come nel programma che segue:

```
NEW
 10 A = 20 : B = 15 : C = 9
 20 T = A + B + C
 30 PRINT "(premere SHIFT e CLR/HOME) CON GRUPPI DI "A" E
    "B" E "C" LA STANZA CONTERRÀ "T" PERSONE
 40 IF T > 75 THEN 100
 50 PRINT "(premere CURSOR UP/DOWN) NON CI SONO PROBLE-
    MI PER OSPITARLI"
 60 END
100 PRINT "È PIÙ DI QUANTO LA STANZA POSSA CONTENERE!"
```

Usando i valori assegnati in riga 10, il calcolatore stamperà quanto compreso fra virgolette nelle righe 30 e 50, poi smetterà di eseguire i comandi quando raggiungerà l'istruzione END. Salterà oltre il comando che lo rimanderebbe a riga 100 dalla riga 40 : THEN 100. Se modificate la riga 10 di modo che il totale superi 75:

```
10 A = 19 : B = 23 : C = 37
```

e fate girare il programma di nuovo, il computer eseguirà il comando dopo THEN e salterà le righe 50 e 60 per eseguire l'istruzione PRINT di riga 100.

Per indirizzare il Commodore 64 a un numero di riga da una istruzione IF-THEN, potete anche includere la parola di comando opzionale GOTO. La revisione della riga 40:

```
40 IF T > 75 THEN GOTO 100
```

produce gli stessi risultati della riga precedente, ma può essere più facile da leggere e da capire.

Per istruire un calcolatore ad agire attraverso un programma basato sui valori che incontra, o su comandi immediati all'interno di un'istruzione IF-THEN, potete modificare il modo in cui esso esegue un programma. Indirizzando il calcolatore da un'istruzione IF-THEN a un numero di riga precedente, è possibile fargli ripetere comandi precedenti con eventuali variazioni.

SIMULAZIONI

Un'istruzione IF-THEN con un numero di riga è come un dito puntato in avanti o all'indietro. Questa capacità di indicare diversi andamenti di azione può essere particolarmente utile nella simulazione di situazioni diverse. Potete applicarla, per esempio, al problema di riempire un locale per una festa con ospiti in arrivo a gruppi senza superare la capienza dell'ambiente.

In una simile situazione, ci sono tre valori da tenere presenti: la capienza della stanza, che potete chiamare variabile C; il numero di ospiti alla festa, che potete chiamare G, e il ritmo al quale arriveranno i nuovi invitati ad ogni viaggio, chiamato R. Se volete sapere con quanti arrivi riempirete la sala senza renderla affollata, potete usare l'istruzione IF-THEN per indirizzare il calcolatore a considerare arrivo dopo arrivo, e segnalare poi quando si verifica il viaggio con cui si raggiunge definitivamente la capienza massima della sala.

Potete iniziare questo nuovo programma con un'istruzione che definisca le dimensioni del locale, il numero di invitati all'inizio della simulazione e il ritmo dei nuovi arrivi:

```
NEW
10 C = 200 : G = 20 : R = 6
```

Il calcolatore predisporrà lo schermo ed evidenzierà lo spazio disponibile con questi comandi:

```
20 PRINT "(premere SHIFT e CLR/HOME)"
30 PRINT "DOPO" N "VIAGGI" G "PRESENTI E SPAZIO PER" C – G
```

In riga 30, la variabile N servirà da contatore per il numero di arrivi attraverso il quale il programma ha istruito il calcolatore. Considerate N contatore e tenete aggiornato G, il numero di invitati, mediante questi comandi:

```
40 N = N + 1 : G = G + R
```

L'ordine cruciale è quello che istruisce il calcolatore a continuare o smettere di considerare gli arrivi:

```
50 IF G < C THEN 30
```

Questa riga dice al computer o di ritornare alla sequenza di riga 30 (se G è minore di C), o di proseguire alla riga finale che vi segnala che è stato raggiunto il limite:

```
60 PRINT : PRINT " ..... RIEMPITO DOPO VIAGGIO" N "←←←←"
```

Quando lo fate girare, questo programma dirà al calcolatore di ripetere quanto contenuto tra virgolette in riga 30 con valori diversi finchè venga raggiunto il limite di capienza della sala. La figura 4.1 illustra i passi seguiti dal programma.

Per prima cosa, il calcolatore immagazzina i valori di C, G ed R (riga 10) nella sua memoria, quindi cancella lo schermo e mette in posizione il cursore per l'istruzione PRINT di riga 30. Quando incontra quell'istruzione, il valore N è stato posto automaticamente a zero e rappresenta la situazione del locale prima dell'arrivo dei primi ospiti:

```
DOPO 0 VIAGGI 20 PRESENTI E SPAZIO PER 180
```

Figura 4.1 — Risultato di un programma di simulazione.

Quando incontra riga 40, il calcolatore considera il viaggio numero 1, che fa aumentare il numero degli invitati a 26. Poi, a riga 50, confronta il numero degli ospiti con la capienza della sala e, vedendo a questo primo passaggio che il numero è inferiore, segue l'istruzione IF-THEN di riga 30. Agendo ancora su riga 30, il calcolatore illustra i valori di N e G dopo questo primo viaggio:

DOPO 1 VIAGGI 26 PRESENTI E SPAZIO PER 174

Il computer continua poi a prendere in considerazione il secondo viaggio di riga 40 e ripete il procedimento. Stampa la stessa proposizione ad ogni viaggio fino al passaggio in cui il numero di ospiti raggiungerà la capienza del locale, quando N è 30. A questo punto, incontrando il comando IF-THEN, il calcolatore va a riga 60 scrivendo il messaggio che la sala sarà da considerarsi riempita con quel viaggio.

Potete aggiungere un dispositivo molto comodo a questo programma istruendolo a evidenziare le condizioni indicate in riga 10:

70 **LIST** 10

Figura 4.2 — Un programma di simulazione che mostra le condizioni che lo hanno guidato.

Questo comando stamperà la riga 10, come illustrato in figura 4.2. Potete ora ristampare la riga per indicare un locale più grande, C, un diverso numero di invitati già presenti, G, o un diverso ritmo di arrivi, R.

Sotto la direzione di un comando IF-THEN che lo rimanda indietro, il calcolatore utilizza le sue capacità ripetitive. Agendo sotto altri comandi il computer può essere mandato in ogni direzione ad altri comandi ancora, da una riga di programma all'altra. Si possono formare itinerari complessi tra istruzioni di programma, ognuna delle quali chiede al calcolatore di fare qualcosa di diverso.

CAPITOLO 5

CONTROLLO DI PROGRAMMA

Essendo una macchina servile, il calcolatore può accettare ed eseguire grandi quantità di comandi, e lo farà nel suo modo, ottuso, ma meticoloso. Con queste istruzioni potete far lavorare la macchina su cose che a voi possono sembrare troppo noioise, ripetitive, precise, lunghe o che richiederebbero troppo tempo. Quando riceve ordini di un linguaggio che riconosce, il vostro calcolatore è in grado di ripetere un dato compito all'infinito e con la frequenza da voi desiderata; può trattare parole o numeri, separandoli o riunendoli, può effettuare i calcoli più scomodi o può produrre display sullo schermo di una tale complessità che altrimenti richiederebbero giorni e giorni di lavoro manuale.

UN CORREDO DI COMANDI PER IL PROGRAMMATORE

Gli esseri umani sono imperfetti: cambiano idea, fanno errori e sono inclini a dimenticare. I programmi che ideiamo sono soggetti alla nostra natura imperfetta, e in considerazione di ciò è possibile contrassegnare, fermare, far ripartire i programmi che usate. È possibile prendere a prestito istruzioni utili da altri programmi e ridisegnarle nella forma voluta.

REM: il taccuino del programmatore

Se riunite molti programmi, troverete comodo un comando che marchi le righe di programma senza influenzare l'azione del calcolatore. Questo comando è REM, che significa nota (in inglese "remark"). Potete apprezzare la sua utilità esaminando il seguente programma a simboli grafici che propone il disegno di una figura quadrettata simile a quella illustrata in figura 5.1. La figura viene ottenuta disegnando prima le linee verticali in bianco nelle colonne aventi numeri pari (nelle righe da 20 a 60) e quindi dividendole con

linee orizzontali in nero (righe da 70 a 120). Da notare che la riga 70 usa il tasto CLR/HOME senza SHIFT, per mandare il cursore "home" (a casa) senza cancellare lo schermo.

```
NEW
 10 PRINT "(premere SHIFT e CLR/HOME)"
 20 A = 0
 30 PRINT TAB (A) "(premere C = e +) (premere SHIFT e CURSOR
    UP/DOWN)"
 40 A = A + 2 : IF A < 25 THEN 30
 50 PRINT
 60 B = B + 1 : IF B < 20 THEN 20
 70 PRINT "(premere CLR/HOME)"
 80 M = 0.
 90 PRINT TAB (M) "(premere SPACE BAR) (premere SHIFT e
    CURSOR UP/DOWN)"
100 M = M + 1 : IF M < 25 THEN 90
110 PRINT "premere CURSOR UP/DOWN)"
120 V = V + 2 : IF < 20 THEN 80
```

Non è immediatamente chiaro per tutti cosa fa questo programma. Invece di farlo girare nel calcolatore ogni volta che volete vedere quello che sta facendo, potete identificarlo con una riga indicante "remark" che, essendo una nota per chiunque legga il programma, non viene elaborata dal calcolatore. Per contrassegnare così il programma, aggiungete una riga che inizi con REM:

```
 5 REM UN PROGRAMMA PER IL COMMODORE 64 CHE DIA UNA FI-
   GURA QUADRETTATA.
```

Quando il calcolatore incontra questa riga REM in un programma, considera il comando come se ciò significasse: "Vai avanti alla prossima riga; quello che segue è una nota per la persona che leggerà il programma". Anche se un programma può sembrarvi perfettamente sensato quando lo preparate, alcune settimane più tardi, se non addirittura pochi minuti dopo, esso può apparire come uno sconcertante assortimento di comandi se non lo avrete contrassegnato con una nota.

Poichè il comando REM può essere messo in qualunque punto del programma, se lo desiderate potete usarlo per descrivere sezioni o righe di comando individuali. Per esempio, nel caso del programma quadrettato, potete

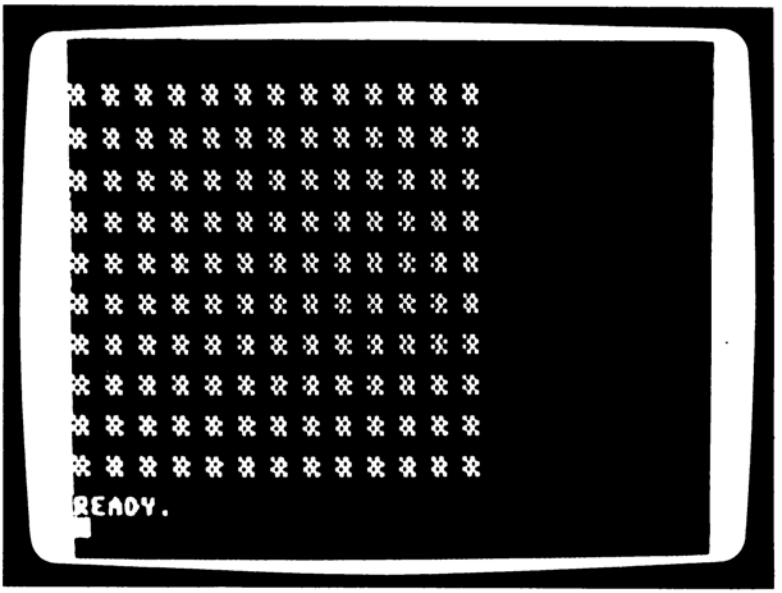

Figura 5.1 — Una figura quadrettata ottenuta con un programma.

aggiungere note prima delle istruzioni che ordinano al calcolatore di produrre righe verticali e orizzontali.

15 **REM** LE SEGUENTI CINQUE ISTRUZIONI DISEGNANO RIGHE VERTICALI IN CARATTERI COLORATI
65 **REM** LE SEGUENTI SEI ISTRUZIONI DISEGNANO RIGHE ORIZZONTALI IN COLORE DI SFONDO

Un programma così contrassegnato sarà più facile da identificare e modificare in seguito. Anche le istruzioni REM sono soggette al limite di lunghezza di 80 caratteri come le altre istruzioni di programma; la riga 65 è lunga quanto può esserlo una riga REM.

Interruzione di un programma in corso di elaborazione

Mentre questo programma sta girando, potreste decidere il volere una figura leggermente diversa. Per fermare il programma si possono usare tre comandi.

Il modo programmato è quello di includere un comando appropriato all'interno del programma stesso. Il comando che interromperà l'elaborazione è STOP. Potete inserirlo strategicamente per fermare il corso del programma

dopo che sono state disegnate le righe verticali dalle istruzioni 20 - 60, in questo modo:

65 **STOP**

Con l'aggiunta di questa riga, il calcolatore si fermerà quando incontrerà l'istruzione informandovi che è stata ordinata un'interruzione (break) forzata mediante la risposta:

BREAK IN 65

Il comando STOP è diverso da END in quanto il calcolatore in questo caso risponde con un messaggio. STOP può essere parafrasato come: "Stampare BREAK IN e il numero di riga e attendere istruzioni dalla tastiera". Un vantaggio del comando STOP è che potete, a vostro comodo, comandare al calcolatore di procedere con il programma con il seguente comando:

CONT

che può essere così parafrasato: "Continuare a svolgere i comandi alla riga di programma che segue l'interruzione". Naturalmente, se il comando STOP viene eliminato dal programma (nel nostro caso togliendo riga 65) il programma stesso continuerà a girare senza interruzioni.

Il secondo modo di fermare un programma viene direttamente dalla tastiera, con il comando che imita STOP in un modo immediato e non programmato. Per arrestare un programma in qualunque punto del suo lavoro, bisogna premere il tasto RUN STOP. Dopo aver ricevuto quest'ordine di tastiera il calcolatore risponderà con il messaggio "BREAK IN". Una volta fermato lo svolgimento con il comando RUN STOP, il calcolatore riprenderà a svolgere i comandi di programma quando riceverà il comando CONT. STOP, inserito in un programma, e il comando RUN STOP inviato dalla tastiera, influenzano l'azione del calcolatore allo stesso modo.

È possibile usare l'uno o l'altro per far fermare il calcolatore a riga 50 dell'attuale programma, proprio dopo che sono stati disegnati i bordi inferiori delle righe verticali per variare eventualmente la figura. Un modo di modificare il display è quello di selezionare un altro carattere o un altro colore per le file disegnate da riga 70 a 120. Anche uno schermo in bianco-e-nero potrà dare variazioni ombreggiando i colori.

Ora che avete interrotto il programma ..... Interrompere un programma può darvi a volte delle idee per variazioni molto interessanti. Per esempio

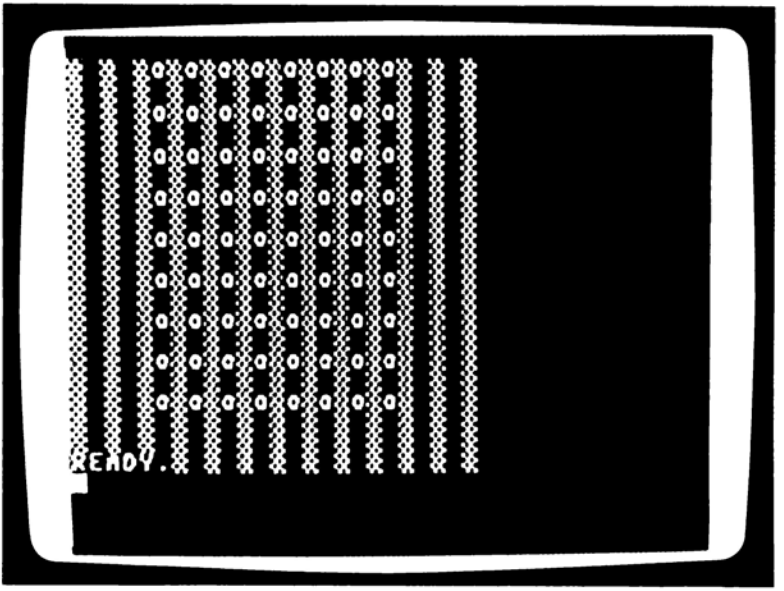

Figura 5.2 — Risultato della revisione del programma che aveva dato il disegno di figura 5.1.

potete cambiare la riga 120 in modo da disegnare colonne di spazi solo fino a riga 18:

```
120 V = V + 2 : IF V < 18 THEN 80
```

Potete creare file orizzontali di cerchi con questa revisione:

```
90 PRINT TAB (M) "(premere SHIFT e W) (premere SHIFT e CUR-
   SOR UP/DOWN)"
```

Ovviamente, è possibile continuare a rivedere la figura fino a trovare quella che più vi piace. Forse questa revisione delle righe 80 e 100 darà il tocco finale:

```
 80 M = 5
100 M = M + 2 : IF M < 20 THEN 90
```

Questi tre cambiamenti produrranno un programma che, una volta avviato, darà la figura di linea e cerchi illustrata in figura 5.2.

Il comando RUN STOP-RESTORE

Il terzo e più invadente modo di fermare un programma in corso ordina al calcolatore di abbandonarlo completamente.

Esso viene mantenuto in memoria, ma senza lasciare alcuna nota. Per arrestare lo svolgimento di un programma con questo comando di interruzione di emergenza, che viene effettuato dalla tastiera, dovete premere assieme i tasti RUN STOP e RESTORE.

Poiché il calcolatore dimentica la sua posizione nel programma, il comando CONT in questo caso non può essere utile per riprendere la sequenza. È possibile rallentare un programma o qualunque risposta del calcolatore premendo il tasto CTRL e tenendolo schiacciato. Quando lo lascerete, il calcolatore ritornerà alla velocità normale.

Abbiamo visto l'intera serie di comandi a vostra disposizione per ristrutturare i programmi esistenti in linguaggio BASIC. Ci sono comunque altri comandi per creare i programmi stessi.

COMANDI PER MUOVERSI TRA LE ISTRUZIONI

Il comando GOTO

Abbiamo visto alcuni comandi che portano il Commodore 64 al di fuori del suo corso abituale attraverso numeri di riga più alti, ma ce ne sono degli altri. Uno, il GOTO, è persino più semplice di IF-THEN. L'abbiamo già usato in una versione del comando di IF-THEN-GOTO, ma è anche possibile usare il comando GOTO da solo per indirizzare il calcolatore a qualche istruzione precedente, ancora e ancora, crando un **"loop"** (anello, iterazione, ciclo) senza fine di azioni. Ideale per compiti ripetitivi, questo tipo di ciclo può essere sospeso solo con uno dei tre metodi di interruzione descritti nel paragrafo precedente.

Il programma seguente comanderà al calcolatore di produrre una tavola di moltiplicazione e divisione "senza fine". (Potreste però anche raggiungere un numero maggiore di quello che il Commodore 64 può gestire ed ottenere così un messaggio di OVERFLOW ERROR (errore di eccedenza). Il numero scelto da moltiplicare e dividere in riga 10 è cinque.

```
NEW
10 A = 5
20 PRINT "(premere SHIFT e HOME) ..... IL NUMERO" A " ....."
30 PRINT "N" TAB (5) "MOLTIPLICATO PER N" (TAB) (22) "DIVISO
   PER N"
```

```
40 PRINT.
50 N = N + 1
60 PRINT N TAB (19) A * N TAB (25) A/N
70 GOTO 40
```

Il comando finale, in riga 70, ordina la ripetizione del programma a partire dalla riga 40. Senza l'istruzione GOTO, il programma avrebbe semplicemente elaborato tre righe di stampa — il titolo di riga 20, il sottotitolo da riga 30 e il primo passaggio al numero di contatore N (quando passa da 0 a 1), la moltiplicazione di A per N e la divisione di A/N da riga 60. Comunque, quando il calcolatore termina con i comandi di riga 60 e traduce il comando GOTO 40 di riga 70, torna indietro alla sequenza di riga 40. Quindi segue automaticamente i numeri di riga crescenti fino a che incontra di nuovo la riga 70, che lo porta fuori sequenza nuovamente a riga 40. Ogni volta che il calcolatore effettua le istruzioni di riga 50 (N = N + 1), viene dato alla stampa un nuovo valore, moltiplicazione e divisione di riga 60.

L'instancabile calcolatore effettuerà i comandi tra le righe 40 e 70 ancora e ancora finché voi non lo fermerete e con i comandi RUN STOP oppure RUN STOP-RESTORE dalla tastiera (o, naturalmente, togliendo l'alimentazione).

Anche se potete ordinare al calcolatore di saltare a una riga di numero più alto, non c'è vantaggio a comportarsi così, poichè gli ordini delle istruzioni di intervento verrebbero perse per sempre. Anche se un secondo comando GOTO rimanderebbe il calcolatore a una riga precedente che era stata saltata, questo metodo di programmazione offre un controllo che potrebbe essere raggiunto più semplicemente da un unico comando GOTO in grado di controllare il tipo di iterazione vista prima.

In questo programma potete assegnare qualunque valore ad A in riga 10 e poi, se state cercando un valore particolare, osservare i relativi valori calcolati così come essi appaiono sullo schermo. Se per esempio volete sapere quando la divisione scende al di sotto di un certo valore, una variazione del comando GOTO può ordinare al calcolatore di fare quella ricerca per voi. Questo comando è strutturato in modo piuttosto rigido, ma facile da usare una volta che abbiate compreso il suo tracciato. Attraverso il comando ON-GOTO, il computer viene istruito a confrontare simboli variabili con una gamma di numeri incorporati, poi viene mandato a una riga di programma. Funziona in modo molto simile all'istruzione IF-THEN-GOTO, tranne per il fatto che la condizione della prima metà del comando è predeterminata a intervalli numerici: uno o più, ma meno di due; due o più, ma meno di tre e così

via. Gli ordini nella metà del comando GOTO puntano a numeri di riga che voi ordinate secondo questi intervalli prestabiliti.

Se volete che il computer vi segnali quando per esempio il valore della divisione A/N scende al di sotto di due, tre o quattro, potete aggiungere un comando dopo riga 60, così:

```
65 ON A/N GOTO 100, 200, 300
```

che dice al calcolatore di valutare A/N, confrontarlo con gli intervalli incorporati e quindi procedere con la riga di programma appropriata - 100, 200 o 300 - in relazione a quel valore. In questo caso potete mettere istruzioni PRINT partendo dalle righe di programma 100, 200 e 300 per mettervi sull'avviso circa i valori prodotti dalle righe precedenti del programma e A/N. Se per esempio non vi interessano i valori al di sotto di due, potreste dare il seguente comando per segnalare e arrestare il programma:

```
100 PRINT TAB (20) "VALORI MINORI DI DUE"
 10 STOP
```

che stamperà "VALORI MINORI DI DUE" nella colonna A/N e quindi farà arrestare il calcolatore con il messaggio "BREAK IN 110".

Allo stesso modo, se volete che il computer continui a svolgere il programma e semplicemente vi segnali quando il valore A/N slitta al di sotto di quattro, e poi ancora al di sotto di tre, potete aggiungere istruzioni partendo da 300 e 200, rispettivamente:

```
300 PRINT TAB (20) "(premere CONTROL e G) (premere CONTROL
    e G) VALORI MINORI DI QUATTRO"
310 GOTO 40
```

e

```
200 PRINT TAB (20) "(premere CONTROL e G) (premere CONTROL
    e G) VALORI MINORI DI TRE"
210 GOTO 40
```

Durante l'esecuzione di questo programma, ognuna di queste coppie di istruzioni segnalerà l'apparire del valore che cercavate per poi riportare il calcolatore nel ciclo moltiplicazione - divisione di riga 40.

È possibile usare il comando ON-GOTO per ricercare e agire su valori specifici. Ponendo le variabili entro una forma che rientri tra gli intervali di valore alquanto particolari della prima metà di questo comando, potete indi-

rizzare il calcolatore verso elaborazioni diverse dai normali svolgimenti sequenziali di un programma. È possibile precisare azioni su qualsiasi intervallo di numeri. Si possono usare istruzioni per allineare numeri, ognuno accoppiato con un intervallo particolare, per far lavorare il calcolatore su più istruzioni. Il comando ON - GOTO segue questo schema: ogni numero di riga stampato dopo la parola GOTO corrisponde in passi agli intervalli progressivamente crescenti del valore della variabile dopo la parola ON.

Potete usare un'istruzione come:

ON A/N **GOTO** 100,200,300,400,500

e così via, per coprire valori sempre maggiori di A/N.

RIPETIZIONI LIMITATE

Potete voler ripetere un insieme di comandi utili, magari non all'infinito (come con l'istruzione GOTO), non condizionatamente (come con l'istruzione IF - THEN), ma semplicemente per un determinato numero di volte. Il Commodore 64 dispone allo scopo di un paio di comandi. Uno tiene conto delle ripetizioni, l'altro riporta il calcolatore al comando iniziale. Questi comandi sono FOR e NEXT. Potete dividerli, uno prima di una serie di istruzioni che desiderate vedere ripetute, l'altro dopo.

Ecco un programma al quale potete aggiungere questi due comandi in un gran numero di applicazioni per ottenere gran varietà di risultati. In questo caso farà disegnare al calcolatore delle frecce.

```
NEW
10 PRINT "(premere SHIFT e CLR/HOME)"
20 X = 5
30 PRINT TAB (X) "(premere SHIFT e M)"
40 PRINT TAB (X) "(premere SHIFT e Z)"
50 PRINT TAB (X) "(premere SHIFT e N)"
```

Attraverso il valore che stampate per X come numero di colonna in riga 20, potete porre la freccia disegnata con le istruzioni delle righe 30, 40 e 50 in qualunque posizione orizzontale. Ora, per disegnare la freccia diverse volte in posizioni differenti, il calcolatore ha bisogno di istruzioni per ripetere le righe 30, 40 e 50 per valori differenti di X, così che lo stesso modello venga rappresentato graficamente da diversi punti di partenza.

Potete dare queste indicazioni con la prima istruzione della coppia di parametri. In questa istruzione voi stabilite un valore di partenza, un valore finale e, (facoltativo), le dimensioni dei passi che il calcolatore deve fare per andare da un valore all'altro. In questo programma grafico potete specificare un numero di colonna rappresentato da X che dovrà venir modificato ad ogni passaggio così che la freccia sia disegnata in ognuna delle diverse posizioni lungo una singola riga, spaziata di 8 colonne, mediante l'istruzione:

25 **FOR** X = 1 **TO** 30 **STEP** 8

L'altro membro di questa coppia di comandi indirizzerà automaticamente il calcolatore alla riga di programma dove si trova il comando FOR-TO-STEP:

55 **NEXT** X

Per ordinare al calcolatore di porre ogni successiva freccia sulla stessa linea orizzontale di quella disegnata in precedenza, dovete aggiungere un'istruzione che rinvii il cursore alla riga di partenza. Tale istruzione sarà:

27 **PRINT** "(premere SHIFT e CURSOR UP/DOWN) (premere SHIFT e CURSOR UP/DOWN) (premere SHIFT e CURSOR UP/DOWN) (premere SHIFT e CURSOR UP/DOWN)".

Quando fate girare questo programma, tutti i comandi posti fra le istruzioni FOR-TO-STEP e NEXT alle righe 25 e 55 verranno ripetute fino al raggiungimento del limite stabilito dopo la parola TO, come illustrato in figura 5.3.

In questo caso vengono ripetute le righe 30, 40 e 50 (comandi del disegno) e la riga 27 (il comando di posizione della fila). Il computer comincerà a disegnare ad ogni quattro posizioni lungo una singola fila in cima allo schermo. La prima freccia parte dalla posizione stabilita da TAB di 1, e l'ultima a TAB di 25. Il valore di X = 1 assegnato nell'istruzione FOR-TO-STEP sostituisce quella iniziale a partire dalla riga 20, ed ogni successivo valore di X sostituisce il precedente.

Usando la coppia di comandi FOR-TO-STEP, NEXT avete ora un programma che disegna una fila di frecce. Con un'altra coppia di comandi FOR-TO-STEP, NEXT su valori di Y potete espandere il programma per disegnare tale gruppo di frecce su più file, e riempirne lo schermo. Per ripetere la prima fila di frecce, la seconda coppia di comandi dovrà porre tra virgolette le istruzioni che disegnano la fila, inclusa la coppia FOR-TO-STEP e NEXT per i valori di Y. Così le frecce a ogni posizione X possono essere disegnate per differenti posizioni Y (file). Per indirizzare il calcolatore verso il basso allo scopo

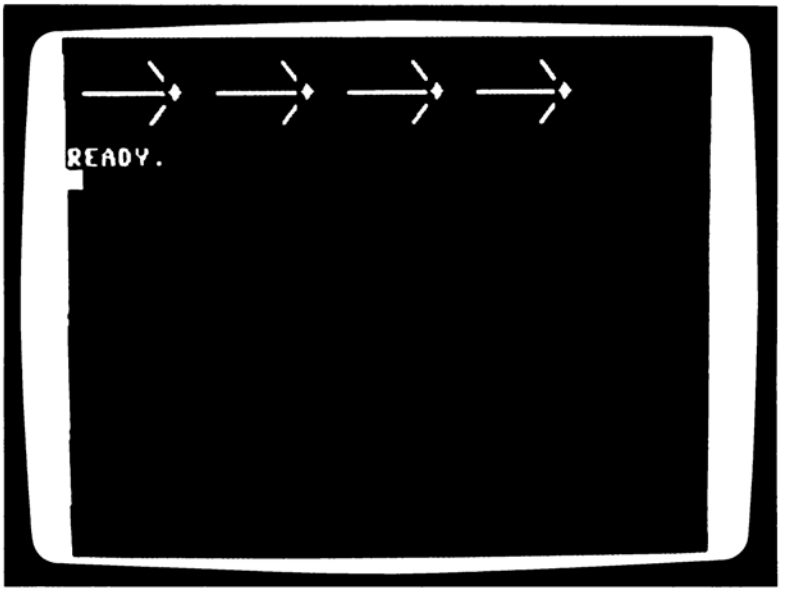

Figura 5.3 — Una fila di frecce ripetute disegnate dal programma.

di disegnare un'altra fila, potete usare un comando che muove il cursore come segue:

```
24 PRINT "(premere CURSOR UP/DOWN) (premere CURSOR
   UP/DOWN) (premere CURSOR UP/DOWN) premere CURSOR
   UP/DOWN)"
```

E per dire al calcolatore di ripetere il disegno in quattro file successive potete aggiungere un'altra coppia di comandi ripetitivi, così:

```
22 FOR Y = 1 TO 4
60 NEXT Y
```

Quando il calcolatore incontra un comando FOR-TO senza la parte STEP, esso ammette un valore STEP di 1 per difetto. Il programma ora in memoria ordina al calcolatore di ripetere i comandi che creavano la prima fila di frecce, di contare quattro righe verso il basso dalla prima fila e disegnare una fila di frecce, quindi scendere di altre quattro righe e disegnare una fila di frecce, e così via fino a raggiungere la quarta fila. Allora il calcolatore smet-

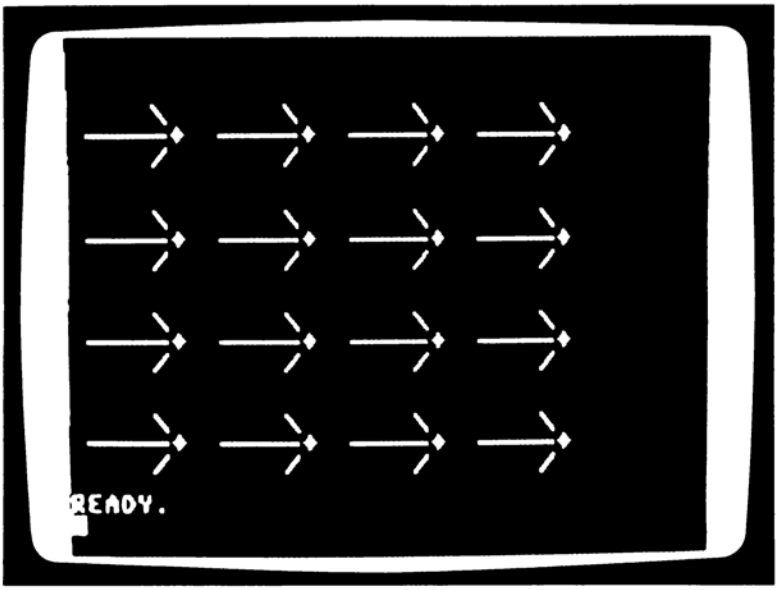

Figura 5.4 — Come riempire lo schermo di frecce modificando il programma di figura 5.3.

terà di disegnare e quattro file di quattro frecce ciascuna riempiranno lo schermo, come in figura 5.4. Poiché l'effetto di riga 20 (che assegnava un valore a X) è stato cancellato da riga 25, potete eliminare questa istruzione in quanto superflua:

20

(Se ora battete LIST, vedrete i comandi incuneati FOR-TO-STEP e NEXT come illustrato in figura 5.5). La prima istruzione NEXT, in riga 55, dice al calcolatore di agire su riga 25, dove è cambiato il valore di X.

Il calcolatore può effettivamente distinguere un ordine di istruzione NEXT da un altro anche se non contrassegnate ogni istruzione NEXT con la variabile della sua riga FOR-TO-STEP; le righe 55 e 60 potrebbero essere scritte così:

55 **NEXT**
60 **NEXT**

Comunque, potrete rintracciare più facilmente i raggruppamenti FOR-TO-STEP, NEXT contrassegnandoli.

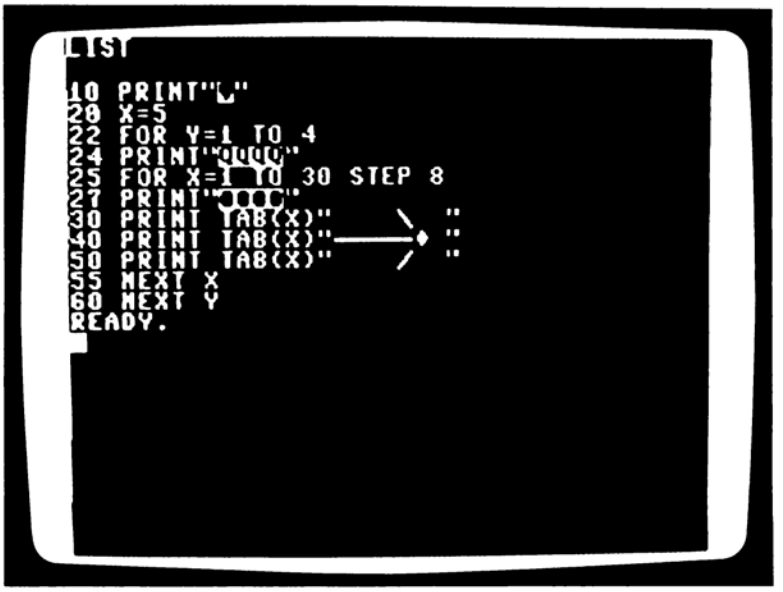

Figura 5.5 — Programma che dà come risultato uno schermo riempito di frecce.

È possibile individuare l'iterazione "incuneata" X FOR-TO-STEP, NEXT all'interno del ciclo Y FOR-TO-STEP, NEXT. Quando un ciclo è incuneato in un altro programma, i comandi dell'iterazione interna sono eseguiti per primi, tante volte quanto viene indicato nel comando FOR-TO-STEP, al primo valore della iterazione esterna, poi attraverso il secondo valore dell'iterazione esterna, e così via. Ecco un riassunto di come questo programma comanda il calcolatore.

In riga 10 lo schermo viene cancellato e il cursore viene posto all'angolo superiore sinistro. In riga 22 il calcolatore si prepara a sostituire un valore di 1 per ogni Y che incontra. In riga 24, il cursore viene fatto scendere di quattro righe in preparazione del disegno. In riga 25 il calcolatore si prepara a sostituire un valore di 1 per ogni X. In riga 27 il cursore viene fatto muovere verso l'alto di quattro file verticali. Gli effetti del movimento verticale nelle righe 24 e 27 entrano particolarmente in gioco nel porre le frecce disegnate una dopo l'altra. Si effettuano le sostituzioni alle righe 30, 40 e 50 e i caratteri stampati risultanti sono disegnati dalla posizione TAB(1). Tutto ciò produce la prima freccia sullo schermo.

Da riga 55, il calcolatore è rimandato indietro all'istruzione FOR-TO-STEP di riga 25, dove viene aggiunto un valore 8 all'ultima X, e il nuovo valore di X viene sostituito alle righe 30, 40 e 50. Il calcolatore disegna la seconda freccia dalla colonna numero 9 della prima fila.

Quando poi incontra riga 55, il calcolatore è rimandato nuovamente all'istruzione FOR-TO-STEP di riga 25 e si ripete il procedimento con un valore di X uguale a 17 sostituito nei comandi TAB.

Tutto ciò continua fino a che venga superato per X il valore di 30. Questo si verifica dopo che è stata disegnata la quarta freccia, quando l'istruzione NEXT X di riga 55 manda il calcolatore alla riga seguente, numero 60. La riga 60 manda il computer all'istruzione FOR-TO-STEP di riga 22, dove viene aggiunto 1 al valore di Y. Con Y = 2 il calcolatore allora si muove su riga 24, che fa andare il cursore quattro righe più in basso.

Poi, incontrando l'istruzione FOR-TO-STEP di riga 25, il computer riaggiusta il valore di X a 1 e procede con riga 27. Questa istruzione mantiene ogni serie di quattro frecce di diverso valore TAB(X) sulla stessa fila. In sua assenza, il calcolatore avanzerebbe automaticamente alla linea di stampa seguente dopo aver disegnato una freccia ad ogni posizione TAB(X) ed ogni freccia apparirebbe separatamente su file diverse.

Svolgendo i comandi da riga 25 a 55 come prima, il calcolatore disegna una seconda fila di frecce, identica alla prima, ma sotto di essa. Dopo che è stata disegnata l'ultima freccia della seconda fila e X supera 30, il calcolatore viene mandato all'istruzione NEXT passando attraverso le righe 55÷60, il che lo rimanda a riga 22.

Le istruzioni FOR-TO-STEP, NEXT Y di riga 22 e 60 impegna allora il calcolatore sulla terza ed infine sulla quarta fila di frecce fino a che i valori X e Y (30 e 4) vengono entrambi superati. Così, vi rimane uno schermo con quattro file, ognuna di quattro frecce. Come potreste sospettare, se due iterazioni incuneate di istruzioni FOR-TO-STEP, NEXT possono venire accumulate insieme nel calcolatore, deve essere possibile riunirne tre o più. Con la stampa grafica del Commodore 64 potete usare questa funzione per creare immagini in movimento di colori mutevoli.

In tutti i programmi, un ciclo controlla tutte le istruzioni ivi contenute, inclusa qualunque quantità di altre iterazioni.

Se si aggiungono comandi che stampano spazi vuoti su una freccia prima che sullo schermo ne venga disegnata un'altra, la prima sembrerà cancellata e la seconda comparirà da sola, fino a che anch'essa scomparirà dallo schermo, sostituita da quella che segue. Il risultato sarà un'elementare forma di animazione.

Per far sì che il calcolatore disegni a turno ogni freccia, dovete aggiunge-

re una serie di tre istruzioni, all'interno del ciclo X FOR-TO-STEP, che stampi spazi su ogni freccia, così:

```
52 PRINT TAB (X) "(premere SPACE BAR 8 volte)"
53 PRINT TAB (X) "(premere SPACE BAR 8 volte)"
54 PRINT TAB (X) "(premere SPACE BAR 8 volte)"
```

Per riposizionare i caratteri vuoti sulla stessa fila in cui la freccia era stata disegnata, potete far precedere un'istruzione che posizioni il cursore, così:

```
51 PRINT "(premere SHIFT e CURSOR UP/DOWN 3 volte)"
```

Ora, quando farete girare il programma, i risultati saranno uguali a prima, solo che ogni freccia scomparirà dopo essere apparsa e verrà disegnata la successiva. Il risultato sarà una freccia che corre da sinistra a destra da una fila all'altra.

Un altro uso intelligente degno di nota dell'iterazione FOR/NEXT è quello che rallenta l'immagine, e che può essere utile in un programma in cui si voglia avere tempo per valutare gli effetti del programma. Avrete probabilmente già notato che il calcolatore, per veloce che sia, impiega un certo tempo per effettuare le sue operazioni. Più è complicato il compito, più tempo impiega. Sapendo ciò, potete sfruttare questo ritardo a vostro vantaggio, controllandolo. Potete aggiungere un ciclo che non contenga comandi attivi reali. E anche se in tale iterazione non vi dovessero essere comandi da ripetere, il calcolatore diligentemente andrà fino in fondo, impiegando del tempo per interpretare ed elaborare i comandi FOR-TO-STEP e NEXT.

Per rallentare il programma di animazione, allora, potrete aggiungere un ciclo vuoto (detto "ciclo di tempificazione") dopo la riga di posizionamento degli spazi vuoti, così:

```
51 PRINT "(premere SHIFT e CURSOR UP/DOWN) (premere SHIFT
   e CURSOR UP/DOWN) (premere SHIFTe CURSOR UP/DOWN)
   (premere SHIFT  e CURSOR UP/DOWN)": FOR T = 1 TO 200 :
   NEXT T
```

Ancora una volta, la parte del comando STEP è facoltativa, e se la tralasciate come è stato fatto qui, il calcolatore automaticamente prende il passo di 1.

In questo programma il calcolatore fa una breve pausa prima di ogni cancellazione mentre lavora inutilmente attraverso l'iterazione vuota. Questo tipo di istruzione FOR-TO, NEXT può essere inserita ovunque per rallentare o congelare l'azione in corso di svolgimento.

Eliminando intelligentemente le istruzioni è a volte possibile creare effetti alquanto diversi dal programma originale. Per esempio, togliendo da questo programma l'iterazione Y FOR-TO, NEXT e associando il comando di posizionamento, è possibile creare un display che mostra una freccia che scende diagonalmente attraverso lo schermo. Togliendo le righe 22,27 e 60, così:

22
27
60

si otterrà tale programma.

CAPITOLO 6

GESTIONE DI PAROLE E INFORMAZIONI

Per sua natura, il calcolatore capisce come trattare i numeri, che infatti indirizzano tutti i suoi comandi operativi. Ma anche senza comprenderne il significato, il Commodore 64 — con l'aiuto della programmazione — può gestire parole e informazioni in qualunque modo esse vengano stampate, elaborandole e creandole dietro vostre istruzioni. Il calcolatore può accettare informazioni in molti modi diversi, e perfino fermarsi nel bel mezzo dello svolgimento di un programma per riceverne. Parole, numeri e simboli grafici possono tutti essere oggetto di manipolazione.

STRINGHE: PAROLE DAL CALCOLATORE

Così come può trattare numeri, il Commodore 64 può gestire parole e caratteri di tastiera stampandoli, separandoli e rimettendoli assieme.

Ovviamente, come essere umani, abbiamo un vantaggio sul calcolatore, poiché comprendiamo circa 40.000 parole più del magro lessico del Commodore 64, inferiore a cento comandi. Ma, alla pari di un giornalaio cieco che può maneggiare le riviste che non sa leggere, il Commodore 64 può elaborare migliaia di parole che non capirà mai.

Malgrado il calcolatore possa gestire caratteri di tastiera — parole, numeri e altri simboli — in vari modi complessi, esso raggruppa questi simboli in due sole categorie. Quando incontra una serie di parole e simboli racchiusi tra virgolette, tratta questo gruppo di caratteri come il carico di un autobus da portare in giro come gli ordinate voi, l'organizzatore. Mentre i caratteri scritti senza virgolette vengono interpretati come comandi, da cui il calcolatore prende ordini, come un taxista carica passeggeri dall'angolo di una strada prendendo ordini da loro.

Un gruppo di caratteri racchiusi tra virgolette viene chiamato **"stringa".** Si possono adoperare simboli variabili al posto di stringhe, come per i numeri. Il Commodore 64 identifica il simbolo di uguale come un comando che assegna una variabile sia a parole, sia a numeri e riconosce queste stringhe at-

traverso un simbolo di variabile "aggiunto". Una variabile intesa come stringa è contrassegnata dal simbolo del dollaro ($), che rappresenta una stringa di caratteri, piuttosto che una variabile numerica. Dato che il computer tratta gli spazi come caratteri, un gruppo di parole separate da spazi verrà elaborato dal Commodore 64 come una lunga stringa di parole.

Potete ordinare al calcolatore di sostituire una frase (o qualunque gruppo di caratteri) ovunque incontri una variabile di stringa in un'istruzione:

```
NEW
10 A$ = "WORDS WITHOUT THOUGHTS NEVER TO HEAVEN GO."
```

Potete allora usare la variabile di stringa, A$, in sostituzione della stringa stessa, quando date il comando successivo:

```
20 PRINT A$
```

Quando viene fatto girare questo breve programma e si incontra riga 20, il calcolatore stampa le parole di Shakespeare sullo schermo allo stesso modo in cui stamperebbe un numero assegnato a una variabile numerica.

Combinare le stringhe: il segno più

Malgrado le stringhe non possiedano alcuna delle caratteristiche numeriche per la cui elaborazione il calcolatore è stato costruito, nondimeno esse possono essere oggetto di calcolo e manipolazione. Usando variabili di stringa potete ordinare al calcolatore di combinare stringhe, troncarle o derivare da esse valori numerici.

Il computer tratta variabili di stringa in combinazione per quanto tratta numeri nell'addizione. In questo caso, il risultato finale sarà una stringa più lunga, piuttosto che un numero più grande. Per aggiungere una stringa all'altra, usate il segno più (+) come comando al calcolatore di produrre una nuova stringa, una combinazione di stringhe su entrambi i lati del segno più.

Aggiungendo altre due stringhe a questo programma, potete predisporre il calcolatore a combinarle, assegnandole alle variabili di stringa:

```
12 B$ = "WILLIAM SHAKESPEARE"
13 C$ = "AMLETO, ATTO III, SCENA 3"
```

Per vedere queste stringhe stampate in combinazione, potete riscrivere riga 20 così:

```
20 PRINT A$ + B$ + C$
```

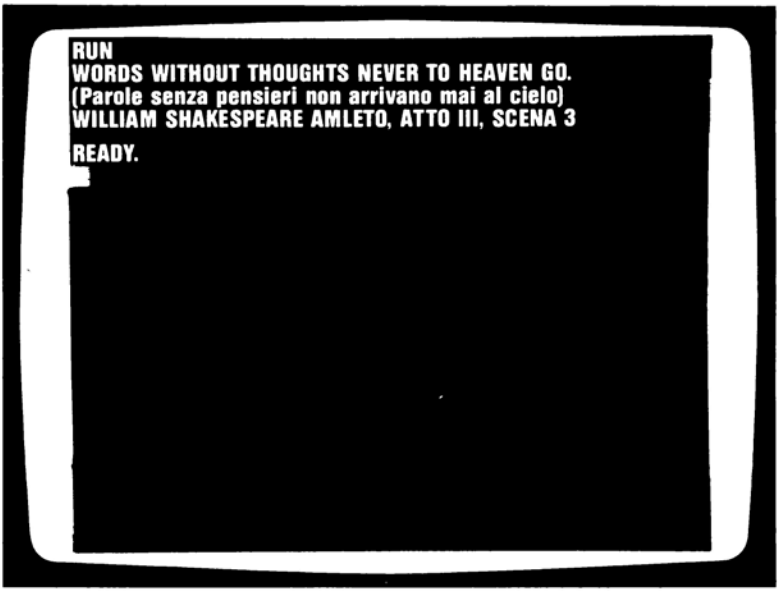

Figura 6.1 — L'effetto di un programma che aggiunge stringhe di parole.

Quando il programma verrà svolto, il calcolatore risponderà col display illustrato in figura 6.1.

Contare i caratteri nelle stringhe: LEN

È possibile estrarre un valore numerico da una stringa con un comando che conta i numeri di caratteri al suo interno, compresi gli spazi. Tale comando può essere utile in qualunque programma che manipoli le stringhe. Il comando che ordina al calcolatore di contare i caratteri si chiama LEN ( ). In un programma potete assegnare una valore di lunghezza di stringa a una variabile numerica con un'istruzione come questa nuova versione di riga 20:

20 A = **LEN** (A$)

Questa riga può essere parafrasata come: "Porre il valore della variabile numerica A uguale al numero di caratteri di stringa A$". Essa ordina al calcolatore di sostituire il numero di caratteri di stringa A$ ovunque incontri il simbolo A nello svolgimento di un programma. Potete allora usare questo valore in un'altra istruzione, così:

30 **PRINT** "A QUOTE" A "CHARACTERS LONG BY" B$ ("Una citazione lunga" A "caratteri di")

ed aggiungere anche un'istruzione per evidenziare la citazione dopo una spaziatura di un paio di righe:

```
40 PRINT : PRINT : PRINT A$
```

Dividere le stringhe: LEFT$, RIGHT$, MID$

Tre comandi dicono al calcolatore di prelevare caratteri dall'interno di una stringa. Usandoli è possibile separare parole o gruppi di caratteri per poi ricostruirli secondo il modello da voi preferito. I tre comandi che frazionano una stringa sono simili nella forma ed ognuno di loro ne toglie un segmento.

Uno di questi comandi, LEFT$, dice al calcolatore di togliere un dato numero di caratteri dalla sinistra di una stringa. I caratteri vengono contati a partire dalla sinistra delle virgolette.

Se voleste una nuova stringa, D$, formata dai primi cinque caratteri della citazione, dovreste chiamare e ordinare la sua formazione mediante il comando:

```
40 D$ = LEFT$ (A$,5)
```

È allora possibile evidenziare la nuova stringa col comando:

```
50 PRINT D$
```

Quando viene fatto girare, il programma produce il display illustrato in figura 6.2. È possibile creare nuove stringhe di lunghezza crescente (fino all'intera lunghezza di A$) partendo dalla stringa originale, includendo il comando LEFT$ in un ciclo FOR con una variabile di contatore:

```
35 FOR I = 1 TO A
40 D$ = LEFT$ (A$,I)
60 NEXT I
```

Il programma ora "racchiude" l'istruzione di frazionamento della stringa di riga 40 e il comando PRINT a riga 50 in una iterazione FOR—TO—NEXT che conta da 1 fino alla lunghezza totale della citazione originale. Quando il programma viene fatto girare, il calcolatore stamperà una fila dopo l'altra; ognuna inizia con il primo carattere della stringa ed è più lunga di un carattere rispetto la precedente. Quando viene raggiunto l'ultimo ciclo, (quando I = A), a D$ verrà assegnato il valore di LEFT$ (A$, A), cioè l'intera lunghezza di A$. D$ sarà uguale ad A$ e l'ultima riga stampata sarà l'intera frase, come illustrato in figura 6.3.

```
RUN
UNA CITAZIONE LUNGA 42 CARATTERI DI WILLIAM SHAKESPEARE
WORDS

READY.
```

Figura 6.2 — Effetto del comando LEFT$.

```
WORDS WITHOUT THOUGHTS N
WORDS WITHOUT THOUGHTS NE
WORDS WITHOUT THOUGHTS NEV
WORDS WITHOUT THOUGHTS NEVE
WORDS WITHOUT THOUGHTS NEVER
WORDS WITHOUT THOUGHTS NEVER
WORDS WITHOUT THOUGHTS NEVER T
WORDS WITHOUT THOUGHTS NEVER TO
WORDS WITHOUT THOUGHTS NEVER TO
WORDS WITHOUT THOUGHTS NEVER TO H
WORDS WITHOUT THOUGHTS NEVER TO HE
WORDS WITHOUT THOUGHTS NEVER TO HEA
WORDS WITHOUT THOUGHTS NEVER TO HEAV
WORDS WITHOUT THOUGHTS NEVER TO HEAVE
WORDS WITHOUT THOUGHTS NEVER TO HEAVEN
WORDS WITHOUT THOUGHTS NEVER TO HEAVEN
WORDS WITHOUT THOUGHTS NEVER TO HEAVEN G
WORDS WITHOUT THOUGHTS NEVER TO HEAVEN GO
WORDS WITHOUT THOUGHTS NEVER TO HEAVEN GO.

READY.
```

Figura 6.3: — Effetto del programma che crea stringhe di lunghezza crescente mediante il comando LEFT$.

Aggiungendo un'altra istruzione potete indicare la fonte della citazione:

```
70 PRINT C$
```

È possibile trasformare questo programma in un test di identificazione rallentando la stampa con un ciclo For-TO-NEXT vuoto, così:

```
38 FOR T = 1 TO 500 : NEXT T
```

Ora, con qualsiasi citazione assegnata ad A$, potete svolgere il programma fermandolo tramite tastiera con un comando RUN STOP e facendolo avanzare mediante il comando CONT.

Il secondo comando di frazionamento si stringa, RIGHT$, ordina al calcolatore di isolare frazioni di stringhe partendo dalla destra. Il nostro attuale programma stamperà le frazioni di A$ calcolate dall'ultima lettera della stringa se il seguente comando di frazionamento a destra sostituisce quello di sinistra in riga 40:

```
40 D$ = RIGHT$ (A$, I)
```

Guidato dalla medesima iterazione FOR, il calcolatore stamperà ora ogni riga con un carattere in più della precedente, ma partirà dal segno di virgolette sulla destra, come illustrato in figura 6.4.

Il terzo comando di frazionamento di stringa, MID$, dice al calcolatore di ritagliare una sezione dal centro di una stringa. Per usare questo comando, includete il nome della stringa da cui deve essere eliminata una parte, la posizione del carattere più a sinistra (calcolato dal lato sinistro della stringa) e il numero di caratteri (calcolato e tolto a destra). Il comando prende la forma MID$ (A$, s,n) dove s è il punto di partenza ed n è il numero di caratteri della frazione. Se, per esempio, doveste togliere la parola di otto lettere THOUGHTS, a sedici spazi dalle virgolette di sinistra, dalla citazione di riga 10, il comando di eliminazione dovrà essere MID$ (A$,16,8). Se inserite questo comando in sostituzione degli altri di nuovo a riga 40, il calcolatore produrrà un'altra lista di stringhe, questa volta in aumento verso l'esterno dal centro della stringa originale, A$, in entrambe le direzioni.

Per usare il comando che dia stringhe che si allarghino **simmetricamente** partendo dal centro della stringa originale, comunque, è necessaria un po' di aritmetica. Potete controllare la lunghezza della stringa con la variabile I, che è il valore dato dall'istruzione FOR-TO-NEXT, come nei comandi LEFT$ e RIGHT$. Il calcolatore trova il mezzo della stringa A$ dividendo la sua lunghezza totale (come in riga 20:A=LEN (A$)) a metà: per A/2 e la metà del

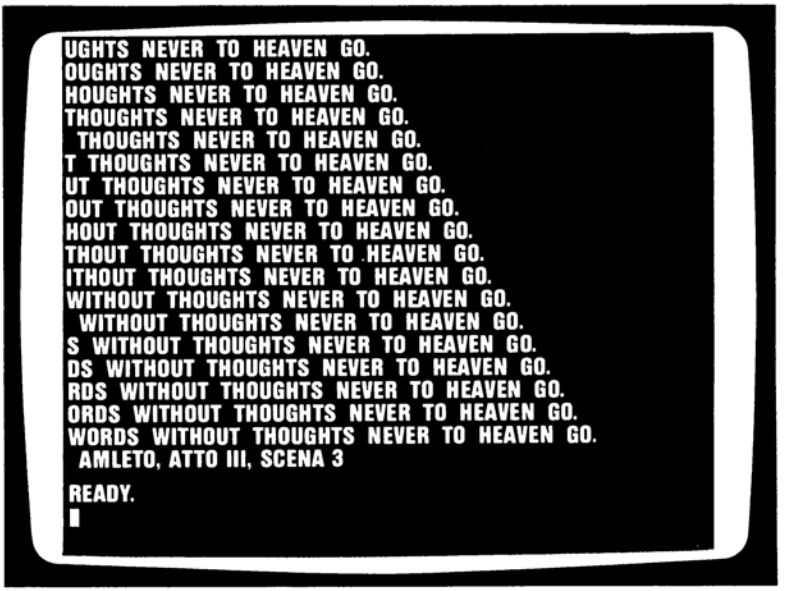

Figura 6.4 — Effetto del programma che crea stringhe di lunghezza crescente mediante il comando RIGHT$.

numero di caratteri, I, da stampare ad ogni passaggio, per I/2. Metà dei caratteri verrà stampata dal lato destro e metà dal sinistro, partendo dal centro. Dato che il carattere all'estrema sinistra di una stringa ha il numero 1, il limite sinistro può essere specificato come A/2 - I/2 + 1. Così potete ordinare al calcolatore, tramite il comando MID$, di produrre stringhe a partire dal centro della citazione originale, A$, con il seguente comando in riga 40:

```
40 D$ = MID$ (A$, A/2-I/2+1,I)
```

Avrete così una lista di stringhe simmetricamente crescenti, come in figura 6.5.

INFORMAZIONI INTERNE

Allo stesso modo di attori versatili in una rappresentazione, i simboli di variabili di numeri e caratteri possono avere molti ruoli diversi all'interno di un programma. Il segno di uguale indirizza queste variabili nei loro rispettivi

ruoli del momento. Ma il Commodore 64 riconosce un altro modo di assegnare i valori alle variabili, che è in un certo senso il modo in cui gli attori di un teatro ricevono la propria parte, assieme. Potete assegnare valori a diverse variabili contemporaneamente mediante una coppia di comandi, DATA e READ, dove stampate gli elenchi ("list") dei valori da inviare al calcolatore e le variabili a cui sono assegnati, rispettivamente. È possibile mandare al calcolatore qualunque numero di valori con il comando DATA, che tratta la lista di informazioni. Potete usarlo, mediante il comando seguente, per mandare i valori di una stringa e due numeri, CHOCKEBERRY 15 e 30:

```
NEXT
10 DATA CHOCKEBERRY,15,30
```

Poi usate il comando che fa le assegnazioni della lista dei valori, READ, per assegnare questi valori rispettivamente alle variabili N$, A e B con questo comando:

```
20 READ N$,A,B
```

Ora potete usare queste assegnazioni, che sono l'equivalente dei comandi:

```
N$ = CHOCKEBERRY
A = 15
B = 30
```

in un'istruzione PRINT come questa:

```
30 PRINT N$ "DOPO" A "MESI"
```

Aggiungete ora le righe 10,20 e 30 a una serie di comandi in cui un valore, B, viene usato per controllare un display PRINT:

```
12 PRINT "(premere SHIFT e CLR/HOME)"
40 FOR X = 1 TO B
50 PRINT TAB (X) "(premere C= e +)"
60 PRINT "(premere SHIFT e CURSOR UP/DOWN) (premere SHIFT
   e CURSOR UP/DOWN)"
70 NEXT X.
```

Queste righe dicono al calcolatore di tracciare una barra orizzontale di lunghezza corrispondente al valore di B dato nell'istruzione DATA di riga 10.

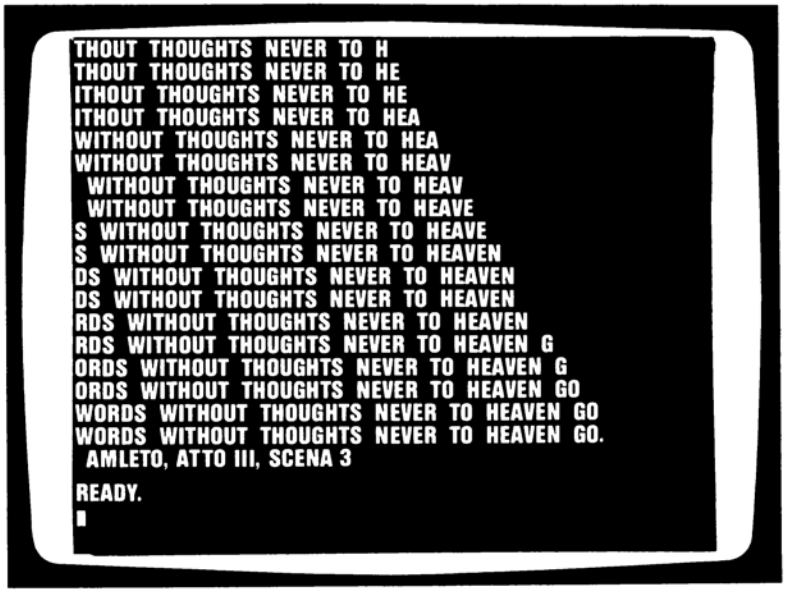

Figura 6.5 — Effetto del programma che crea stringhe di lunghezza crescente mediante il comando MID$.

Facendo girare questo programma, vedrete l'intestazione:

CHOCKEBERRY DOPO 15 MESI

e sotto una barra orizzontale, lunga 30 colonne.

Questi due comandi di gestione d'informazioni, uno che crea una lista di informazioni, l'altro che preleva da questa lista, possono essere usati ovunque all'interno di un programma. Non è necessario che un programma preceda l'altro. Se ponete un'istruzione READ prima di un'istruzione DATA, il calcolatore scoverà l'istruzione DATA quando incontrerà READ, ed effettuerà l'assegnazione dei valori alle variabili. Poichè esso effettua sostituzioni ogni volta che incontra un'istruzione READ, potete usare un'iterazione per ordinare al computer di sostituire diverse serie di valori presi da una istruzione DATA per una singola serie di variabili listate nella sua istruzione di accompagnamento READ.

Questo programma può essere modificato per trattare un gruppo di valori, agendo su ogni gruppo, ritornando poi all'istruzione READ per nuove assegnazioni alle variabili. Potete facilmente aggiungere più nomi e valori all'istruzione DATA:

```
10 DATA CHOCKEBERRY,15,30, VETCH,25,14,BEARDWORT,17,27,
   CHICKWEED, 9,21, THYME, 11,25
```

Ma se fate girare il programma ora, il calcolatore vi risponderà come prima, stampando solo le informazioni relative a CHOCKEBERRY, poiché agisce sull'istruzione READ soltanto una volta. Per farlo lavorare sull'istruzione READ abbastanza volte da sostituire una variabile per ogni serie di informazioni di tre dati nell'istruzione DATA, dovete aggiungere una coppia di comandi FOR-TO, NEXT per ripetere le righe da 20 a 70:

```
17 FOR I = 1 TO 5
80 NEXT I
```

Includendo i comandi READ-DATA in un ciclo FOR, potete ordinare al calcolatore di introdurre valori diversi dall'istruzione DATA in qualsiasi serie di comandi di uso corrente. Per rallentare la velocità di elaborazione e di stampa, è possibile aggiungere un'iterazione vuota prima che il calcolatore passi a un'altra serie di valori:

```
75 FOR T = 1 TO 2500 : NEXT T
```

Infine, poiché il ciclo da riga 17 a 80 include i controlli del cursore (che presentano un normale ritorno di riga dopo che è stata tracciata ogni barra), è necessario aggiungere una istruzione in bianco PRINT:

```
25 PRINT.
```

Quando fate svolgere questo programma il calcolatore cancella lo schermo e fa muovere il cursore all'angolo superiore sinistro. Esso assegna allora alle variabili N$, A e B i primi tre valori dell'istruzione DATA, rispettivamente CHOCKEBERRY, 15 e 30. Il calcolatore stampa poi l'istruzione da riga 30, quindi segue le istruzioni per tracciare la barra orizzontale. Dopo essere passato attraverso il ciclo vuoto FOR-TO, NEXT in riga 75, il calcolatore viene poi rimandato dalla riga 80 all'istruzione READ di riga 20, che assegna alle variabili N$, A e B i successivi tre valori nell'istruzione DATA (VETCH, 25,33) e ripete la stampa e la pausa. Il comando NEXT in riga 80 rimanda di nuovo il calcolatore attraverso l'istruzione READ e un'altra serie di valori. Questo processo continua fino al passaggio finale attraverso l'iterazione FOR-TO, NEXT quando vengono assegnati gli ultimi valori dell'istruzione DATA e il display di figura 6.6 viene completato.

Oltre ad effettuare nuove assegnazioni alle variabili, le strutture di programma DATA, READ, e FOR-TO, NEXT possono anche essere usate per dare una denominazione diversa, contrassegnata da un numero, a ciascun valore di DATA. Mediante un ciclo FOR-TO, NEXT potete ordinare al calcola-

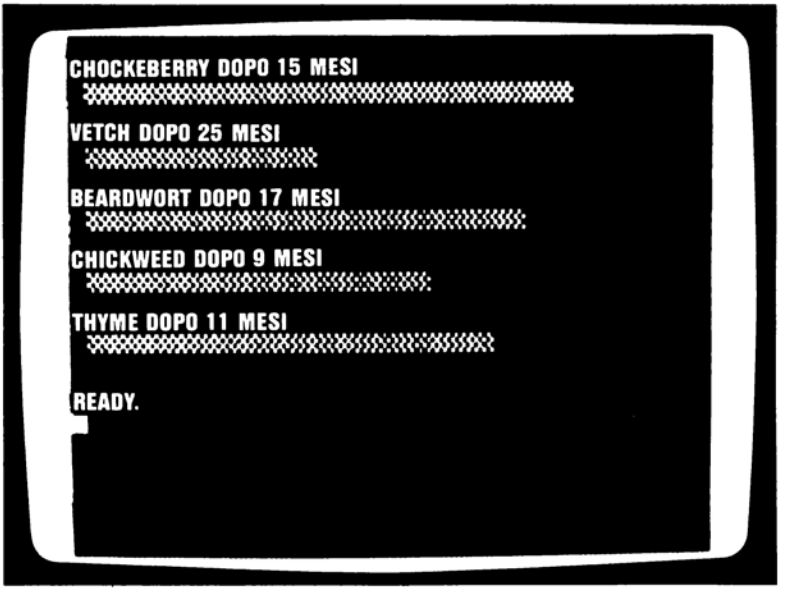

Figura 6.6 — Barre ottenute usando le istruzioni READ e DATA in un ciclo FOR.

tore di numerare ogni variabile ai passaggi attraverso l'iterazione. Per dare un nome alle variabili così che ognuna venga appaiata a un singolo valore, dovete modificare riga 20 così:

```
20 READ N$ (I), A (I), B (I)
```

il che numererà le variabili. Al primo passaggio attraverso l'iterazione, quando I = 1, le tre variabili N$ (1), A (1) e B (1) saranno assegnate a CHOCKEBERRY,15, e 30. Al secondo passaggio, quando I = 2, le tre variabili N$ (2), A (2) e B(2) rappresenteranno VETCH,25 e 33. Il calcolatore ripeterà poi questo procedimento fino all'ultimo passaggio, quando N$ (5), A(5) e B(5) saranno contrassegnati e assegnati. Avrete ora sette serie di variabili **subscritte.**

È possibile ordinare al calcolatore di portarle in un'istruzione di programma per numero, con queste righe:

```
35 PRINT N$ (I) "DOPO" A(I) "MESI"
45 FOR X = 1 TO I
```

Il programma risultante produrrà gli stessi risultati del suo predecessore non subscritto, ma il vantaggio è che è possibile ordinare al calcolatore di

usare queste variabili contrassegnate numericamente in vari modi, a seconda dei numeri calcolati o assegnati in altre istruzioni del programma. Potete per esempio ottenere la scrittura tabulare dei nomi delle piante e dei loro ritmi di sviluppo nel tempo. Ora le variabili subscritte A(I) e B(I) si presentano come una comoda combinazione di istruzioni PRINT e FOR-TO, NEXT che evidenzieranno tutte le informazioni:

```
 85 PRINT
 90 FOR I = 1 TO 5
100 PRINT N$(I) "AUMENTATO A" B(I) "DOPO" A(I) "MESI"
110 NEXT I
```

Il programma ordinerà ora per prima cosa la stampa e i simboli grafici della barra come aveva fatto prima, poi la stampa di un elenco della crescita delle piante, come illustrato in figura 6.7.

Se avete un'applicazione per le variabili con più di una singola marcatura numerica per ognuna, potete aggiungere più marcature. Le variabili N$ (I,J), A(I,J) e B(I,J) possono essere designate sovrapponendo un'altra iterazione FOR-TO, NEXT, che calcoli con la variabile J, attorno ai cicli I. I numeri di J potrebbero per esempio rappresentare l'intervallo di irrigazione di ogni pianta, e potrebbe andare da 1 a 3. È possibile usare così diverse iterazioni per ordinare al calcolatore di designare una serie di variabili nella forma N$(I,J,K,L,M), in cui per esempio I potrebbe essere il numero di codice della pianta, J il suo periodo di irrigazione, K la quantità, L la dose di fertilizzante ed M il tempo concesso.

Ogni ciclo di istruzioni FOR-TO, NEXT che usate per assegnare valori a nomi di variabili subscritte moltiplica il numero delle variabili. Ogni istruzione DATA dovrebbe contenere tanti valori quanti sono richiesti dalla relativa istruzione READ. Quando il computer incontra un comando READ che richiede più informazioni di quante ne fornisca un comando DATA, esso farà arrestare il programma scrivendo il messaggio:

OUT OF DATA ERROR (errore mancanza dati)

Quando il calcolatore incontra per la prima volta una variabile subscritta non numerata, esso automaticamente lascia dello spazio a lato per undici variabili contrassegnate numericamente che potrebbero venire prodotte più tardi. Così facendo, esso riserva spazio nella sua voluminosa memoria per ogni assegnazione — A(1)=, A(2)=, e così via — che il vostro programma potrebbe produrre con i comandi DATA e READ. Le undici variabili saranno numerate da 0 a 10, per ogni serie di subscritti. Così, ovunque il calcolatore

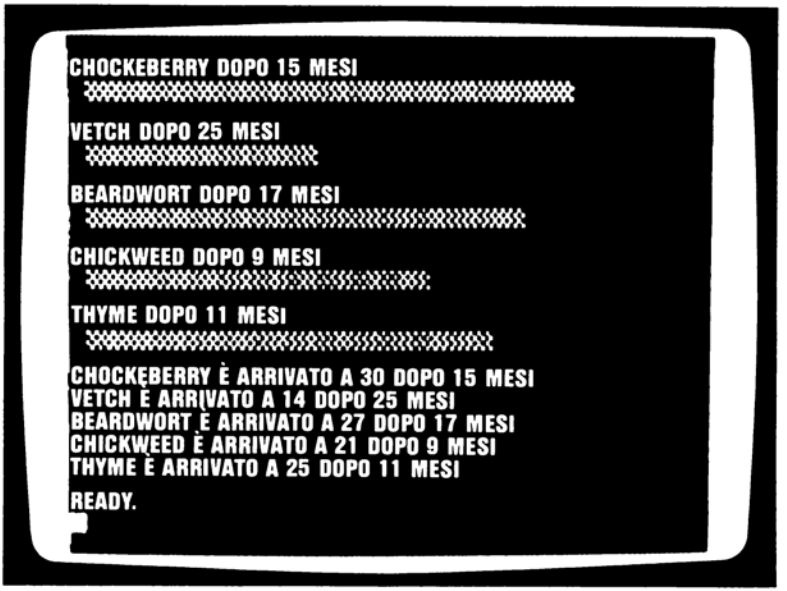

Figura 6.7 — Un altro uso dei valori DATA e READ del programma che ha prodotto la figura 6.6.

incontri una variabile A(I), esso riserverà spazio per le variabili A(0), A(1) fino ad A(10).

Se volete che il calcolatore tratti variabili numerate al di fuori di questa gamma, potete ordinargli di creare loro dello spazio. Con il comando DIM voi date un nome alla variabile, e al subscritto il numero più alto che volete attribuire. Volendo riservare spazio per le variabili da N$(0) a N$(50), per esempio, dovrete inviare il comando:

DIM N$(50)

Se per esempio prendete in considerazione cinquanta persone ognuna delle quali può lavorare su tre turni differenti, e volete avere a disposizione dello spazio per tutte le possibili combinazioni di nomi, orari e prestazioni, le variabili risultanti potrebbero avere qualsiasi forma tra N$(1,1), A(1,1), B(1,1) e N$(50,3), A(50,3), B(50,3). Potete usare il comando DIM per conservare spazio per tutte queste variabili e valori mandando gli ordini:

DIM N$(50,3), A(50,3), B(50,3)

in un'istruzione all'inizio del programma. Incontrando questo comando, il calcolatore metterà da parte un "array" (insieme) di spazi nella sua memo-

ria a partire da N$(0,0) per arrivare a N$(50,3). In un programma che funzioni senza utilizzare tutti gli spazi accantonati da un'istruzione DIM non si verificheranno problemi di spazio, mentre invece il programma affollato di variabili senza spazi di riserva — per esempio un N$(51,3) — verrà fermato nel suo svolgimento da un calcolatore intollerante mediante la risposta:

?BAD SUBSCRIPT ERROR IN (numero di riga)
(errore di subscritto in riga no. ...)

L'OROLOGIO INTERNO

In qualunque momento accendiate il vostro calcolatore, anche senza toccare la tastiera, il Commodore 64 lavora segretamente, tenendo il tempo. Un orologio elettronico interno incomincia a funzionare nello stesso istante in cui date alimentazione e funziona fino a quando non viene tolta la corrente. Si può leggere l'ora, sotto forma di un numero di sei cifre indicanti ore, minuti e secondi, ordinando al calcolatore di rappresentarla sullo schermo. Questo valore è memorizzato, e continuamente aggiornato, con il nome di TIME$. Per controllarlo in ogni momento dovete inviare il comando:

```
PRINT TIME$
```

Il simbolo del dollaro in TIME$ indica che il calcolatore memorizza questo valore, anche se numerico, in quanto variabile di stringa, come una parola. Potete trasformare il calcolatore in orologio con questo breve programma:

```
NEW
10 PRINT "(premere CLR/HOME)"
20 PRINT TIME$
30 GOTO 10
```

che, quando girerà, mostrerà il ticchettio dei secondi in alto sullo schermo. È possibile includere in un programma un confronto IF-THEN che ordina al calcolatore di controllare il valore di TIME$ (che può essere anche scritto TI$) e di agire quando raggiunge un certo valore. Tale istruzione potrebbe essere:

```
IF TIMES = "013520" THEN PRINT "IL VOSTRO TEMPO
È SCADUTO": STOP
```

ed essa, quando verrà trovata dal calcolatore un'ora, 35 minuti e 20 secondi dall'accensione, dichiarerà sullo schermo IL VOSTRO TEMPO È SCADUTO fermando lo svolgimento del programma.

Potrete anche regolare l'orologio sull'ora del momento, come un qualsiasi altro orologio. Per esempio, potrete sincronizzare l'orologio del calcolatore con uno che dia l'ora di mezzogiorno con questa semplice istruzione di assegnazione:

```
TIME$ = "120000"
```

CONTATTO ESTERNO

Anche quando sta lavorando sui comandi di un programma, il Commodore 64 non è interamente insensibile a quanto sta accadendo attorno a lui. Può rispondere a istruzioni ben più sottili e istruttive dell'interruzione controllata che date premendo RUN STOP o togliendo l'alimentazione. Ma il calcolatore, servile e privo di immaginazione, deve essere indirizzato nella giusta direzione prima che possa sapere cosa gli volete dire.

Il comando INPUT

Potete aggiungere un comando che faccia arrestare il calcolatore, prendere le informazioni che avete per lui e poi procedere all'elaborazione utilizzando ciò che gli avete fornito. Questo comando è INPUT. Comandato a questo modo, il calcolatore può interagire con una persona alla tastiera o con apparecchiature cui sia collegato. Il comando INPUT dice al calcolatore di "prepararsi" mettendo da parte una o più variabili e poi "travasare" i valori battuti alla tastiera. Esso quindi procede col programma, sostituendo questi valori come se stesse rispondendo a un comando di assegnazione o a una coppia di comandi DATA, READ.

Usando la semplice forma del comando INPUT, potrete chiedere al calcolatore di dirvi che attende informazioni; esso poi prenderà quello che avrete inserito (premendo il tasto RETURN) per includerlo nel programma. Il comando prende la forma di una semplice richiesta di un valore di variabile. È possibile usarlo per esempio per ottenere l'immediato raddoppio di qualunque numero inviato dalla tastiera con questo breve programma:

```
NEW
10 INPUT A
20 PRINT A *.
```

L'istruzione INPUT A dice al calcolatore di stampare un punto di domanda sullo schermo per segnalarvi che è in attesa di una risposta dalla tastiera. Se non fate nulla il calcolatore resterà congelato nella sua azione, in attesa, su riga 10. Se battete un numero, diciamo 537, e poi premete il tasto RETURN, il calcolatore riceverà quel numero, farà l'assegnazione A = 537 e poi svolgerà i comandi della riga che segue. In questo caso la moltiplicazione 537 x 2 comparirà sulla riga successiva sotto forma di 1074.

Il comando INPUT si traduce così: "Scrivi un punto di domanda, poi prendi il valore successivo inviato dalla tastiera e sostituiscilo con la variabile dell'istruzione che segue". Il comando INPUT dice anche al calcolatore di agire sulle stringhe allo stesso modo, come in questo breve programma:

```
10 INPUT A$
20 PRINT A$ + A$ + A$
```

il che farà ripetere tre volte qualunque dato abbiate inviato.

Allo stesso modo di DATA e READ, un singolo comando INPUT può essere usato per dire al calcolatore di agire sia su stringhe, sia su numeri, così:

```
10 INPUT A$, A
20 PRINT TAB (A) A$
```

Se fate svolgere questo programma e rispondete poi con:

```
DALLA TASTIERA, 20
```

il calcolatore scriverà le parole DALLA TASTIERA iniziando dalla colonna 20.

Potete usare il comando INPUT come DATA e READ, scrivendo i valori dopo il punto di domanda (come dopo un comando DATA) e ponendo variabili per l'assegnazione dopo il comando INPUT (come dopo un comando READ). C'è comunque una differenza: usando un'istruzione INPUT potete inviare nuovi valori durante lo svolgimento di ogni programma senza cambiarne una riga. Un secondo tipo di comando INPUT vi consente di dire al calcolatore di condensare due comandi in uno. Questa forma di comando ordina la scrittura sullo schermo, invece del semplice punto di domanda, prima di aspettare una risposta dalla tastiera. Potete utilizzarlo scrivendo, dopo la parola INPUT e tra virgolette, una domanda o un'istruzione che vorreste vedere comparire sullo schermo, seguita da un punto e virgola per separare le variabili che dite al calcolatore di chiedere.

Dopo aver inviato il comando NEW per cancellare dalla memoria l'ultimo programma, potete costruirne uno partendo da questa forma di comando INPUT, così:

```
10 INPUT "QUAL È IL CODICE PREGO?"; A$.
```

che dice al calcolatore di scrivere la vostra domanda "QUAL È IL CODICE PREGO?" e poi attendere la stringa che può essere sostituita con A$. Questo tipo di istruzione INPUT, anche se sembra migliore, equivale esattamente ai seguenti comandi:

```
10 PRINT "QUAL È IL CODICE PREGO": INPUT A$
```

Aggiungendo un'istruzione condizionale (IF-THEN) potete programmare le risposte scritte del calcolatore in modo che varino secondo le vostre risposte "parola d'ordine" dalla tastiera. Potete predisporre un programma in modo tale che il calcolatore incontri un comando condizionale IF-THEN dopo che un comando INPUT ha inserito informazioni di tastiera nel programma da usare in altri comandi. Agendo così, fate fare al programma uno scambio di informazioni a due vie, una conversazione **interattiva** col vostro calcolatore.

Per creare un tale programma dall'attuale istruzione INPUT, riga 10, dovete aggiungere un comando IF-THEN e due possibili risposte, così:

```
20 IF A$ = "MI HA MANDATO IL VECCHIO MARINAIO" THEN 40
30 PRINT : PRINT "NON PARLO CON ESTRANEI." GOTO 10
40 PRINT : PRINT "OK. TI PARLERÒ QUESTO CALCOLATORE LA-
   VORA SU UN CHIP 6510."
50 PRINT "SONO SOLO UN INTERPRETE DEL CHIP DI ELABORA-
   ZIONE CHE EFFETTIVAMENTE FA SVOLGERE L0"
60 PRINT "SPETTACOLO QUI. QUANDO DAI DEGLI ORDINI È IL
   CHIP 6510 CHE"
70 PRINT "LI FA ESEGUIRE."
```

Quando fate girare un programma come questo, il calcolatore riceve l'ordine di prendere un valore (o in questo caso una stringa di caratteri) dalla tastiera e poi di confrontare quel valore con un altro e di agire in accordo al confronto. Il comando INPUT prende la stringa inviata dalla tastiera e la assegna ad A$. Il comando IF-THEN la confronta con la stringa originale. Se non c'è riscontro, il comando successivo, riga 30, ordina un rifiuto, "NON PARLO CON GLI ESTRANEI", e manda il calcolatore ad agire sull'istruzione INPUT di riga 10, che chiede ancora "QUAL È IL CODICE PREGO?".

Quando la risposta inviata dalla tastiera (e assegnata ad A$) rende la condizione IF vera, il calcolatore verrà mandato dal comando THEN a riga 40, a stampare (PRINT) i segreti più riposti del computer, dopo di che il programma ha termine. Quando le risposte dalla tastiera non trovano riscontro con le condizioni dell'istruzione IF-THEN, comunque, il calcolatore resta intrappolato nell'iterazione di riga 10 attraverso 30, attraverso 10, scrivendo il rifiuto e richiedendo la parola d'ordine più e più volte. Il Commodore 64 è particolarmente ostinato con le istruzioni INPUT. Molti altri programmi possono essere interrotti con il comando RUN STOP, ma non quelli che attendono una risposta a un comando INPUT. Per arrestare tali programmi bisogna usare il comando più energico RUN STOP-RESTORE, che ordina al calcolatore di fermare lo svolgimento del programma mantenendolo però in memoria.

Il comando GET

Ovviamente voi determinate le azioni del calcolatore quando scrivete un programma, ma potete anche far dipendere l'azione dal risultato di un calcolo numerico o da qualche altro valore dato all'elaboratore attraverso un apparecchio esterno. C'è un altro comando, nato dallo stesso bisogno di informazioni del comando INPUT, e che comanda il calcolatore in modo simile, ma più pulito: GET. Tramite questo comando potete ordinare al vostro Commodore 64 di agire su un tasto premuto alla tastiera senza aspettare RETURN. Esso ordina sostituzioni immediate e può ordinare istantaneamente valori di parecchie variabili, ma non scrive messaggi tra virgolette o punti di domanda. Il limite principale di questo comando è che esso prende solo un carattere — numero o stringa — per ogni variabile.

Potete usare il comando in un suo proprio programma, così:

```
NEW
 5 PRINT "(premere SHIFT e CLR/HOME)"
10 GET A$
20 PRINT "(premere CLR/HOME)" A$
30 GOTO 10
```

Quando questo breve programma viene fatto girare nel Commodore 64, il calcolatore assegna il primo tasto premuto alla variabile A$, poi fa muovere il cursore all'angolo superiore sinistro dello schermo e stampa il carattere del tasto che viene premuto. Se si schiaccia più di un tasto, come in questa risposta:

ERRARE È UMANO

il calcolatore prenderà la prima lettera battuta, E, e la scriverà. Poi, per effetto del comando GOTO in riga 30, il calcolatore ritornerà a riga 10 e 20 per leggere e far comparire sullo schermo ogni lettera, una alla volta. Le lettere si sostituiranno a vicenda nell'angolo, immediatamente, poichè il calcolatore legge più in fretta di quanto non possa scrivere una persona. Di fatto, per usare GET, dovrete **sempre** inserire un'iterazione che faccia ripetere continuamente. A differenza di INPUT, un singolo comando GET non aspetterà abbastanza a lungo da permettervi di immettere almeno un singolo carattere. Se volete renderveene conto da soli, cancellate la riga 30 e provate di nuovo il programma.

I dispositivi che rendono il comando GET utile o seccante, a seconda dei punti di vista, sono due: ordina al calcolatore di ignorare qualunque cosa tranne la variabile che gli è stata assegnata ed agisce senza attendere che venga premuto il tasto RETURN.

Il comando ASC

Ogni tasto, quando viene premuto, fornisce informazioni al calcolatore. Questa informazione viaggia come un codice numerico su cui agisce il computer, per esempio facendo comparire la lettera Z sul display dello schermo (premere Z) o interrompendo un programma in corso (premere RUN STOP). Potete vedere il numero inviato da ogni tasto usando un comando che ordina numericamente la lettera di ogni tasto. Questo comando è ASC (" ") e produce il numero di codice del primo carattere della stringa scritta tra parentesi. Per esempio, ASC ("Z") è 90. Potete usare l'istruzione GET in un programma che mostrerà sullo schermo il numero di codice di ogni tasto nel momento stesso in cui lo schiacciate, così:

```
NEW
10 GET A$ : IF A$ = " " THEN 10
10 PRINT A$ TAB(5) ASC(A$)
30 GOTO 10
```

Quando fate girare questo programma, il carattere e il numero di codice di ogni tasto che premete verranno scritti sullo schermo istantaneamente. La riga 10 dice al calcolatore di ricercare il tasto che viene premuto successivamente. Se non ne viene battuto alcuno durante l'attimo che impiega per svolgere una singola operazione GET, allora A$ non ha alcun valore (A$ = " ") e il calcolatore è rimandato indietro a cercare di nuovo. Quando viene battuto un tasto, viene dato un valore ad A$ e il calcolatore va a riga

20, dove riceve l'ordine di stampare il tasto battuto e poi, spaziando, il codice numerico. La riga 30 riavvia l'elaborazione.

Mentre il programma gira, vedrete che ogni combinazione di battitura di tasti (ad eccezione di RUN STOP e RESTORE, che arrestano il funzionamento del programma), produce un numero di codice. Infatti, quattro tasti daranno un numero di codice ognuno, ma nessun valore. Questi quattro, segnati f1 f2, f3 f4, f5 f6 ed f7 f8 sono conosciuti come **tasti di comando** (function keys) e non sono stati inseriti nella tastiera per influenzarne il display, ma per servire da commutazioni programmabili che possono essere usate per segnalare al calcolatore tasti di display disponibili all'uso pur proteggendo gli altri.

Se premete f1 vedrete che esso corrisponde al codice 133, e che SHIFT-f1 (al quale potete pensare come f2) corrisponde a 137. Sapendo ciò potete usare qualsiasi tasto di comando come regolatore per qualche azione, mentre il programma è in corso. Aggiungendo le seguenti istruzioni al programma di cui sopra,potete ordinare al calcolatore di uscire dall'iterazione e di intraprendere un'altra azione:

```
25 IF ASC(A$) = 133 THEN 40
40 PRINT "(premere SHIFT e CLR/HOME)" TIME$
```

Potete usare altri tasti in modo simile. Per esempio per sfruttare ulteriormente il tasto di comando potete aggiungere:

```
27 IF ASC(A$) = 137 THEN 50
50 PRINT "ORA L'HAI FATTO!"
60 GOTO 50
```

Il comando PEEK

Il Commodore 64 può andare al di là della tastiera per avere informazioni su cui lavorare. Fa ciò attraverso i suoi circuiti e non è necessario chiamare un elettricista o spianare un'arma. Infatti, inserendo un paio di spinotti sul fianco del Commodore 64 nella presa targata CONTROL PORT 1 (figura 6.8), potete aprire un'altra linea di informazioni.

Il Commodore 64 ha la capacità di rilevare elettronicamente le posizioni dei paddle o joystick e di interpretare queste posizioni come se fossero numeri, da 0 a 255. Potete per esempio ordinare al calcolatore di prelevare valori dalle posizioni dei paddle con i comandi che li introducono nel programma, PEEK(54297) e PEEK(54298). Poi, agendo come se questi valori fossero stati assegnati in uno dei modi già visti, il calcolatore li elaborerà come fa-

Figura 6.8 — CONTROL PORT 1, la presa per il paddle.

rebbe con ogni altro valore. Il computer può leggere un valore numerico da ogni paddle. Usando il comando PEEK per leggere lo stato elettrico in un paddle, è possibile ordinare al calcolatore di assegnare quel valore a una variabile, su cui quindi si potrà agire nel programma.

I comandi PEEK possono essere usati in un'istruzione come la seguente, che assegna il valore di ognuno dei paddle alle variabili X e Y:

```
NEW
10 X = PEEK(54297) : Y = PEEK (54298)
```

I comandi PEEK (54297) e PEEK (54298) si possono tradurre così: "Generate un numero dal valore elettrico del paddle. Quando questo viene girato completamente in senso antiorario, il valore è 0; quando lo è in senso orario, il valore è 255; generate i valori intermedi a seconda della posizione".

L'istruzione di riga 10 prepara il calcolatore a sostituire il valore, da un paddle per la variabile X, e dall'altro per la variabile Y. Potete trovare qual è uno e qual è l'altro provandoli con un programma simile al seguente. Potete

aggiungere scrittura, cancellazione dello schermo, istruzioni di posizionamento e un comando di ciclo iterativo, così:

```
 5 PRINT "(premere CLR/HOME)"
20 PRINT TAB(10) X TAB (20) Y
30 GOTO 10
```

Quando questo programma è in funzione, i paddle forniscono i valori attraverso i comandi PEEK. La riga 20 del programma ordina al calcolatore di scrivere il valore di ogni posizione del paddle vicino alla parte alta dello schermo. Incidentalmente, i valori al di sotto di 100 sullo schermo saranno di tre cifre, come 990, 980 ecc.

È possibile aggiungere un'istruzione per cancellare il valore precedente e mantenere solo l'attuale valore indicato, così:

```
15 PRINT "(premere SHIFT e CLR/HOME)"
```

Incontrando l'istruzione GOTO, il calcolatore viene rimandato indietro al comando che assegna il valore del paddle alle variabili. Così, i valori scritti sullo schermo stanno al passo con le posizioni del paddle. Mentre questo programma è in funzione, i valori che vedrete sullo schermo vengono prodotti da un dispositivo che agisce come espansione del calcolatore. Questa è l'essenza del controllo del calcolatore.

Potete usare questi valori in un programma così come se usaste valori originati o forniti al suo interno, e con due valori manipolati così facilmente, potete intraprendere ogni genere di azione. Per esempio, è possibile disegnare direttamente con i paddle derivando valori di TAB e di spaziatura verticale dalle posizioni del paddle. Partendo ancora da riga 10 potete usare il seguente programma:

```
 1 PRINT "(premere SHIFT e CLR/HOME)"
 5 PRINT "(premere CLR/HOME)"
10 X = PEEK(54297) : Y = PEEK(54298)
20 H = 30/255 * X : V = 20/255 * Y
30 FOR I = 1 TO V : PRINT : NEXT I
40 PRINT TAB(H) "(premere SHIFT e Q)"
50 GOTO
```

I comandi di assegnazione di riga 20 scalano i valori presi dai paddles alle dimensioni dello schermo. L'iterazione formata dai comandi di riga 30 svolge una funzione di spaziatura secondo il valore di uno dei paddle.

CAPITOLO 7

MEMORIA DI BASSO COSTO — NASTRI

USO DI PROGRAMMI CON CASSETTE MAGNETICHE

La gente viaggia sia in aereo, sia in pullman. L'uno è più veloce e costoso, l'altro è meno comodo e più lento, ma relativamente conveniente. Lo stesso tipo di scelta è possibile per la memorizzazione e l'uso di programmi e file d'informazioni, sia preconfezionati, sia vostri. Un sistema a disco farà muovere programmi e informazioni più velocemente; un sistema a cassette sarà più lento, ma molto più economico.

Potete caricare i vostri programmi su comuni cassette di registrazione a nastro tramite un registratore a cassette Commodore. Si tratta di un registratore collegato in modo speciale (chiamato DATASSETTE TM) che va inserito sul retro del Commodore 64, come illustrato in figura 7.1. Azionandolo secondo le istruzioni che appaiono sullo schermo ogniqualvolta usate il registratore, potete memorizzare tutto ciò che risiede nella memoria di programma del calcolatore su nastri poco costosi.

Il Commodore 64 riconosce comandi semplici per travasare programmi da e sul nastro. Occorre solo premere qualche pulsante sul registratore seguendo le istruzioni sullo schermo. Potete usare la cassetta allo stesso modo in cui registrereste della musica o un discorso. Per utilizzare appieno un nastro, assicuratevi che le tacche di protezione di registrazione al margine della cassetta siano bloccate, che il nastro sia avvolto dall'inizio, mostrando il lato marrone opaco dalla finestrella della cassetta.

Fate scrivolare la cassetta nelle guide di plastica del coperchio del registratore. Chiudete quindi il coperchio e sarete pronti ad usare il nastro per la memorizzazione del programma. La corrente per mettere in moto il registratore viene dal calcolatore stesso. Potete usare il contatore per avere una traccia di dove siano i vostri programmi prendendo nota di programma e numero, proprio come fareste su qualsiasi altro registratore. Questa è una buona idea, dal momento che il sistema di memorizzazione su nastro non creerà automaticamente un indirizzario dei programmi ivi contenuti.

I programmi saranno piuttosto trattati alla stregua di canzoni o discorsi registrati al microfono, uno dopo l'altro.

La procedura per registrare un programma è abbastanza semplice. Se per esempio avete nella memoria del calcolatore un programma che volete conservare sotto il nome ENCHANTMENT, potete inviare il comando:

SAVE "ENCHANTMENT"

Il calcolatore risponderà, a sua volta, con:

PREMERE RECORD & PLAY

Se voi poi premete questi due tasti sul registratore contemporaneamente, il calcolatore risponderà subito con:

OK
SAVING ENCHANTMENT
(sto memorizzando)

Lo schermo sarà svuotato, il motore del registratore girerà e si accenderà una luce rossa. Quando il calcolatore avrà finito di registrare il programma, farà ritornare il display sullo schermo e darà una richiesta READY con il cursore lampeggiante. A questo punto, a meno che non vogliate memorizzare un altro programma, potete premere il tasto STOP. Il programma ora risiede in due posti — sul nastro e nella memoria del calcolatore.

È possibile portare un programma dal nastro nella memoria del computer altrettanto semplicemente. Il calcolatore esaminerà un intero nastro per trovare il nome di quel programma e lo caricherà non appena trovato. Potete ricaricare il programma ENCHANTMENT nel calcolatore da un punto qualunque del nastro prima della collocazione del programma dando il comando:

LOAD "ENCHANTMENT"

Il calcolatore risponderà con le istruzioni:

PREMERE PLAY

Quando fate ciò il calcolatore vi risponderà velocemente con:

OK
SEARCHING FOR ENCHANTMENT
(sto cercando)

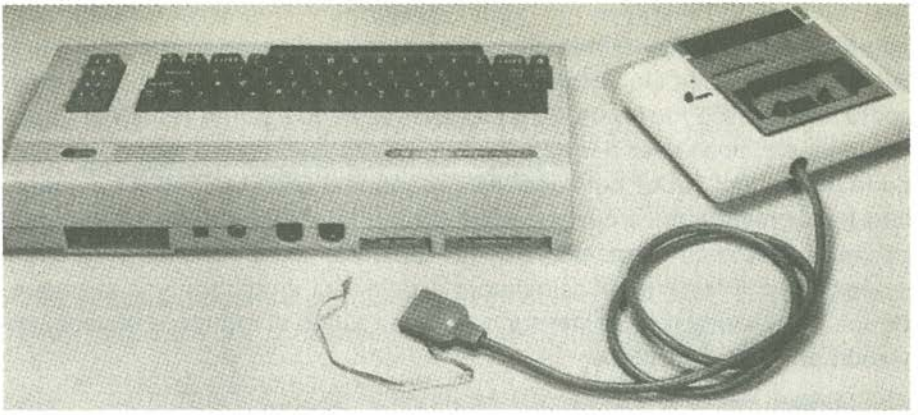

Figura 7.1 — Collegamento tra calcolatore e registratore.

Se qualche programma precede ENCHANTMENT, il calcolatore svuoterà lo schermo e farà una pausa ad ogni programma che incontra, scrivendo il messaggio:

FOUND nome del programma

ma continuerà la sua ricerca di ENCHANTMENT. Se per esempio sono stati registrati prima altri programmi chiamati AGGRAVAMENTO, SFIDA, OSTACOLI, il calcolatore svuoterà lo schermo mentre esegue la ricerca, elencando ogni programma che incontra fino a quando non trovi quello richiesto. Dovrete allora premere il tasto C= per caricare il programma in memoria. Incidentalmente, ogni volta che il calcolatore fa fermare il nastro su un programma durante la sua ricerca, potete caricare quel programma invece di quello che gli avete fatto cercare premendo il tasto C=. Se fate terminare la ricerca al calcolatore, esso poi caricherà ENCHANTMENT non appena lo avrà trovato. Alla fine vedrete questo display:

```
FOUND AGGRAVAMENTO
FOUND SFIDA
FOUND OSTACOLI
FOUND ENCHANTMENT
LOADING
READY.
```

Questo è il modo in cui il calcolatore vi dice di aver incontrato, ma tralasciato, altri tre programmi prima di aver individuato il programma ENCHANTMENT e di averlo caricato nella memoria.

È meglio non "sovrapporre" nomi di programmi. Il calcolatore si fermerà ad ogni nome di programma che contenga, nei suoi caratteri iniziali, il nome del programma che sta cercando. Ciò potrebbe succedere se voi memorizzaste sullo stesso nastro un altro programma sotto il nome ENCHANT e poi comandaste, dopo aver riavvolto il nastro, di caricare "ENCHANT". Trovando le lettere ENCHANT nel nome del programma ENCHANTMENT, il calcolatore si arresterebbe e caricherebbe ENCHANTMENT. Potete usare la capacità di ricerca del Commodore 64 per generare un indirizzo dei programmi su un nastro. Ordinando al calcolatore di ricercare un programma che sapete non potrà trovare, lo avvierete nella sua routine di ricerca e stampa. Inviando un comando come:

LOAD "ZZZ"

per un nastro che non ha tale programma, il calcolatore farà una pausa e scriverà sullo schermo il nome di tutti i programmi che incontra prima di arrivare alla fine del nastro.

FATTI E FILE SU NASTRO

I programmi vanno e vengono sul nastro, ma in realtà essi sono solo ordini per far operare il calcolatore. A volte potete voler memorizzare e trovare informazioni-indirizzi, nomi di località, persone o fatti. È possibile ordinare al sistema nastro-calcolatore di prendere le voci scritte alla tastiera e memorizzarle come file d'informazioni.

Poichè il calcolatore non dispone di un'adeguata serie di ordini per lavorare coi file d'informazioni (come fa coi programmi), avrete bisogno di un programma che gli dica cosa fare. Quello che segue usa tre nuovi comandi — OPEN, PRINT e CLOSE — che controllano il flusso di informazioni tra nastro e calcolatore.

```
100 REM UN PROGRAMMA CHE CREA FILE SU NASTRO
110 DIM A$(100) : INPUT "NOME DEL FILE"; F$
120 OPEN 1,1,2, F$
130 INPUT A$(I)
140 "IF A$(I) = "CHIUDERE FILE" THEN 170
150 PRINT # 1, A$(I)
160 I = I + 1 : GOTO 130
170 CLOSE 1
```

È possibile usare questo programma nel modo seguente: primo, mandate il comando

RUN

Il calcolatore vi risponderà con la domanda:

NOME DEL FILE?

Se scrivete un nome di vostra scelta, diciamo LARVE, e premete RETURN, il calcolatore vi chiederà di attivare il registratore:

PREMERE RECORD & PLAY

Poi, dopo aver fatto ciò, lo schermo si vuoterà, si accenderà la spia rossa del registratore e il nastro comincerà a girare. A questo punto il calcolatore sta ordinando il nome LARVE sul nastro come un nome di file. Quando ciò si è verificato, riapparirà lo schermo con la scritta e le nuove righe:

OK
?

Il punto di domanda è la richiesta da parte del calcolatore per il primo articolo del file. Se scrivete un articolo, come LOMBRICO, e premete il tasto RETURN, il calcolatore vi domanderà l'articolo successivo con un altro punto di domanda.

?

Se scrivete un altro articolo, per esempio BACO DA SETA, e lo inviate col tasto RETURN, apparirà di nuovo il punto di domanda. Questo programma può accettare 100 articoli. (Riga 110 ha creato un "array" (insieme) di 100 variabili per le risposte). Potete inviare articolo dopo articolo fino a quando non avrete inserito la quantità desiderata. A quel punto risponderete al punto di domanda mediante CLOSE FILE, e il calcolatore codificherà su nastro l'intera lista di articoli. Lo schermo si svuoterà, la spia rossa del registratore si accenderà e girerà il nastro. Quando sarà stato caricato l'ultimo articolo il nastro si fermerà, comparirà sullo schermo la richiesta READY e la vostra lista sarà registrata.

A questo modo potete porre su un nastro qualunque numero di file, uno dopo l'altro. Ovviamente, codificati come sono, questi file non vi sono di grande utilità. Ma con un altro programma che comanda il sistema potete e-

strarre gli articoli dal file e mostrarli sullo schermo. Il programma che segue reperirà un file dal nastro per nome ed elencherà ogni articolo sullo schermo.

```
200 REM UN PROGRAMMA CHE REPERISCA UN FILE
    DAL NASTRO
210 DIM A$ (100)
220 INPUT "NOME DEL FILE"; F$
230 OPEN 1,1,0,F$
240 INPUT #1,A$(I)
250 PRINT I TAB (5) A$(I)
260 I = I + 1
270 IF ST > 64 THEN 240
280 CLOSE 1
```

Notate il comando INPUT in riga 240. È una variante di INPUT usato per leggere gli articoli da memoria su nastro o disco invece che dalla tastiera.

Quando fate girare questo programma per vedere gli articoli di un file, come per esempio il file LARVE che abbiamo creato, vedrete apparire la domanda:

NOME DEL FILE?

Una volta scritto il nome del file desiderato e premuto il tasto RETURN, il calcolatore vi darà istruzioni per azionare il registratore:

PREMERE PLAY

Dopo di che lo schermo si svuoterà ed il nastro girerà fino a che non saranno trovati il nome del file e gli articoli. Se fate girare il programma quando il nastro è avvolto dall'inizio e il file è posto verso la fine del nastro, il calcolatore farà passare tutti i nomi fino a trovare quello richiesto. (Se il nome è contenuto in un nome di file più lungo, il calcolatore reperirà questo file invece di quello da voi richiesto. Per esempio verrà reperito il file HAPPENCHANCE, se incontrato per primo, in luogo del file richiesto HAPPEN). Infine il nastro si fermerà e il calcolatore mentre scrive l'articolo sullo schermo indicherà il numero assegnato ad ogni articolo della lista e l'articolo stesso.

Potete usare questi programmi separatamente per memorizzare e radunare file su nastri e se i vostri programmi assegnano agli articoli variabili nella forma A$(I), potete incorporarli.

CAPITOLO 8

MEMORIA VELOCE — DISCHI

Poco dopo la sua comparsa, l'efficiente Pony Express dovette cedere il passo al telegrafo, più veloce e tecnologicamente superiore. Allo stesso modo, in molte applicazioni del calcolatore, il sistema di memorizzazione su nastro di cui abbiamo parlato al capitolo 7 sta cedendo il passo a un nuovo sistema. Progresso tecnologico nei confronti dei sistemi di memorizzazione a nastro, il sistema a disco offre una velocità di accesso che può diventare determinante quando è necessario effettuare frequenti scambi di programmi e informazioni tra calcolatore e strumento di memorizzazione.

UN MATRIMONIO DI MACCHINE

Un sistema a disco opera sotto il controllo elettrico e logico del Commodore 64 come una espansione della memoria del calcolatore stesso, un magazzino di programmi ed informazioni. Ogni elemento di questo sistema può fornire più del doppio della capacità di memoria del Commodore 64, cioè 64 kilobytes. Con un sistema di memorizzazione a disco potete mettere da parte e poi riutilizzare i vostri programmi nonché quelli già scritti per il Commodore 64.

Più complessa del sistema a nastro, la memorizzazione su disco lavora attraverso una scatola delle dimensioni del calcolatore che contiene i suoi propri circuiti e una piccola memoria. Lavorando su istruzioni del calcolatore, questa **unità disco** codifica magneticamente programmi ed informazioni su sottili cerchi di plastica chiamati **dischi,** ognuno dei quali è contenuto in una busta di plastica quadrata. Questi dischi sono rivestiti con un materiale magnetizzabile e rispondono, come i nastri di registrazione, a un dispositivo elettromagnetico che esamina le loro superfici all'interno della scatola di unità disco. L'unità disco include un motore per far girare i dischi all'interno delle loro buste. Poiché questi dischi possono girare in una unità uno alla

volta, per essere poi tolti e sostituiti da altri dischi, essi formano una serie intercambiabile e illimitata di depositi di programmi e informazioni che passano da e per il calcolatore.

La figura 8.1 mostra un'unità disco ed alcuni dischi.

Una volta collegato elettricamente al calcolatore, il sistema a disco risponde anche ai comandi che non fanno parte del vocabolario del Commodore 64. Attraverso questi comandi potete dire al calcolatore di accantonare su un disco un programma o un'informazione dalla sua memoria e di riversare da un disco nella sua memoria.

Il Commodore 64, che può comunicare con un disco alla volta, ospiterà cinque unità-disco, ma è comunque possibile sfruttare convenientemente la maggior parte delle applicazioni con un paio di unità-disco soltanto. Infatti una singola unità-disco sarà sufficiente per la maggior parte delle applicazioni, anche se richiederà da parte vostra più inserzioni ed estrazioni di dischi.

Il sistema a disco e il calcolatore possono scambiarsi i controlli interattivamente. I vostri comandi possono indirizzare il calcolatore a una particolare unità-disco (se ne avete installata più di una) e a un particolare programma o file sul disco dell'unità. All'inverso, un programma che indirizza l'azione del calcolatore da un disco può guidarlo a svolgere comandi o sequenze come fareste voi dalla tastiera.

Ogni programma o file d'informazioni uscito da un disco è alimentato all'interno della memoria del calcolatore prima di intraprendere qualsiasi azione. L'informazione si fa strada tra unità-disco e memoria per mezzo di un semplice cavo elettrico.

Ogni unità-disco contiene i circuiti per far girare il suo motore e far muovere il meccanismo di lettura. Infatti, l'unità-disco ha la sua piccola parte di spazio di memoria per conservare programmi ed informazioni, come pure il suo alimentatore. Non c'è bisogno di un elettricista per i collegamenti. Ecco cosa bisogna fare.

Per prima cosa, togliere la corrente al calcolatore. Senza alimentazione, non è possibile danneggiare la macchina. Il cavo montato sull'unità-disco ha un connettore metallico a 6 poli su entrambe le estremità. Inserite un capo di questo cavo nel retro dell'unità disco nella presa proprio sopra il fusibile. Poi inserite l'altro capo nella presa rotonda più lontana dalla spia di corrente sul lato posteriore del calcolatore. L'ultimo collegamento da fare è la spina a tre poli. Inserite l'estremità a sagoma quadra nel lato posteriore dell'unità-disco e l'estremità standard nella presa di corrente. I collegamenti sono illustrati in figura 8.2.

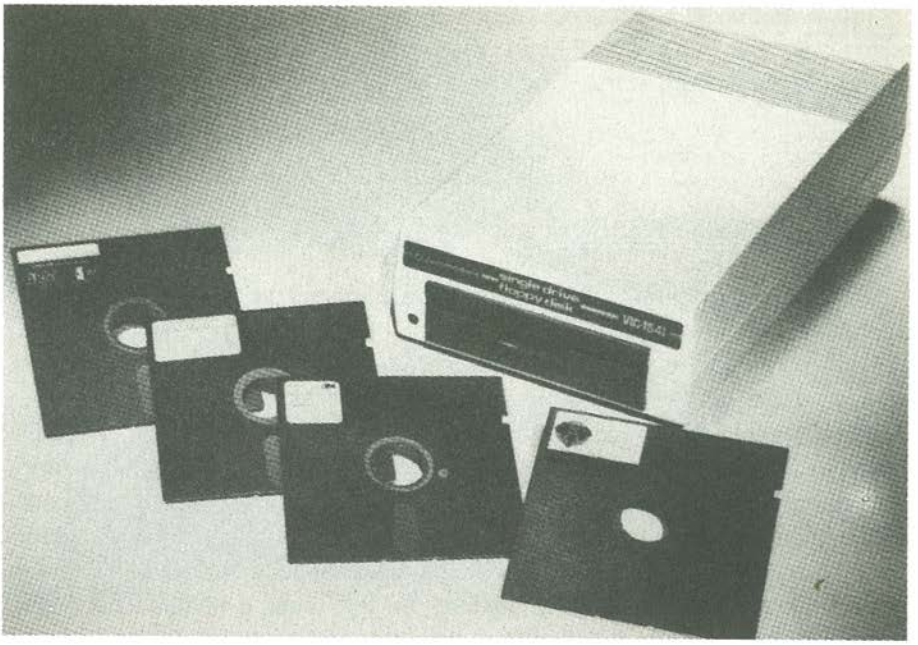

Figura 8.1 — Una unità-disco ed alcuni dischi.

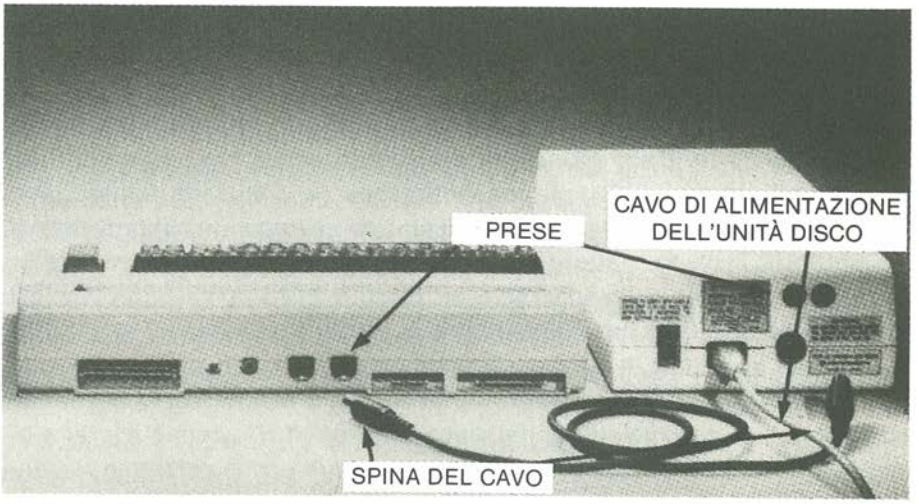

Figura 8.2 — Collegamento tra calcolatore e unità-disco: prese, cavo di alimentazione dell'unità-disco, spina.

Una volta data alimentazione prima all'unità-disco e poi al calcolatore, il collegamento è completo. Potete sistemare gli elementi del sistema — calcolatore, unità-disco e monitor, o televisione — ovunque arriveranno i fili.

USO DI DISCHI PREREGISTRATI

Per prima cosa, qualche parola su come trattare i dischi. Come la carta, i dischi possono essere in bianco o scritti. Potete copiare dischi registrati, scrivere su sezioni vuote di dischi registrati e scrivere sopra, o cancellare, parti di dischi già registrate.

Potete anche proteggere il contenuto di un disco come fareste con della carta scritta posta in una cartelletta, così che non ci si possa scrivere sopra inavvertitamente. La busta quadrata che avvolge il disco protegge la superficie magnetizzabile da cancellazioni o alterazioni nell'unità se questa busta è completamente sigillata su tutti i lati e manca una tacca intagliata, chiamata tacca **write-enable** (scrittura consentita). Se tale tacca è visibile sulla busta, come illustrato in figura 8.3, l'unità-disco sarà distolta dal modo di sola lettura. Potete allora dire al calcolatore di registrare ciò che avete scritto sul disco.

Potete "congelare" tutti i programmi e le informazioni su un disco che porta tale tacca, coprendola con un resistente pezzo di nastro che viene generalmente fornito assieme ai dischi in bianco. Quando i meccanismi dell'unità-disco sentono che lo spazio è coperto, l'unità verrà commutata al modo di sola lettura.

Come segnalato da quasi tutti gli opuscoli, i dischi sono piuttosto fragili. Persino quando un disco ha sopportato qualche infortunio senza riportare danni visibili, la sua superficie magnetizzabile può risultare danneggiata. Campi magnetici, forti variazioni di temperatura, piegature e contaminazioni con sporcizia, polvere, unto e simili sono molto pericolosi. Informazioni e programmi registrati sulla superficie del disco in piste magnetiche invisibili possono venire alterati da un impercettibile graffio o da contaminazioni. Nascosto nella sua busta, quasi tutto il disco è protetto in qualunque momento e potete maneggiare la busta senza riguardo. Ma la testina di trascinamento legge dove la superficie stessa del disco appare attraverso una apertura ovale, e attraverso questa apertura possono passare anche polvere e sostanze contaminanti. È possibile evitare i rischi più comuni prendendo due semplici precauzioni: togliere sempre i dischi dall'unità prima di spegnere l'unità-disco o il calcolatore; scrivere sulle etichette prima di applicarle sulla busta.

Figura 8.3 — La tacca che consente la scrittura.

Il disco e la sua busta di plastica sono contenuti in una copertina di carta che non ricopre completamente la busta. Questa copertina protegge la superficie del disco e la finestrella ovale che altrimenti sarebbero esposti.

La testina magnetica dell'unità-disco, che registra e legge lungo la pista, fa contatto col disco attraverso la finestrella mentre il disco stesso gira all'interno della busta. Poichè questa è quadrata ed ha due facce, ci sono otto modi possibili di inserirla nell'unità; solo un modo però sarà quello esatto.

La testina si trova nella metà posteriore dell'unità, quindi il disco va inserito dalla parte della finestrella. Un lato della busta non è etichettato ed ha un aspetto relativamente incompleto; l'altro è contrassegnato da una etichetta ed è liscio. Se quest'ultimo lato viene inserito nell'unità, il lato magnetizzabile del disco sarà di fronte alla testina. Perciò, inserite il disco in un drive box così: introducete dolcemente il disco fino in fondo, facendo attenzione che il lato con la finestrella sia rivolto verso la macchina con la faccia contrassegnata dall'etichetta rivolta verso l'alto. Potete quindi abbassare lo sportello fino a chiudere l'apertura. Per togliere il disco bisogna sollevare lo sportello fino allo scatto azionato da una molla. Il disco scivolerà fuori per metà per azione della molla stessa.

Inserire il disco nell'unità e chiudere lo sportello completano materialmente il sistema calcolatore-disco, ma a meno che non vengano dati co-

mandi particolari che indirizzino l'unità-disco, il sistema è inutile quanto un bibliotecario analfabeta. Anche quando è materialmente ed elettricamente collegato al sistema a dischi, il calcolatore deve ricevere istruzioni per sapere cosa fare.

Il computer può rispondere ai vostri comandi come pure ai comandi di programmazione BASIC in esso riportati. Le istruzioni, quando sono inserite su disco, compilano automaticamente una lista di programmi e informazioni di file. Il disco TEST/DEMO, fornito con le unità-disco Commodore, contiene già diversi programmi, pur non essendo completamente compilato. Useremo il disco TEST/DEMO per cominciare. Potete comandare una lista di programmi su quel disco (o qualunque altro), e il calcolatore risponderà con un display indicante il nome dato al disco e una lista dei suoi file e programmi. Ognuno è preceduto da un numero che indica lo spazio da esso occupato nel disco.

Potete ordinare al calcolatore di rappresentare questa lista mediante il comando:

LOAD "$", 8

Il calcolatore risponderà con un messaggio sullo schermo, e la spia rossa dell'unità-disco si accenderà per indicare che l'unità è in azione. Potete considerare la spia accesa come un promemoria di non estrarre il disco in quel momento.

Se il sistema calcolatore-unità-disco non riconosce il comando del disco, la spia rossa lampeggerà per segnalare che l'ultimo comando non è stato eseguito. Quindi il sistema accetterà quei comandi che riconosce e farà spegnere la spia dopo averli eseguiti.

Se avete inviato il comando di cui sopra, vedrete comparire il messaggio:

SEARCHING FOR $

Il simbolo del dollaro, $, è un'abbreviazione dell'indirizzario. Una volta trovatolo, il calcolatore risponderà così:

LOADING

Ciò significa che il computer sta copiando informazioni dal discco (in questo caso la lista dell'indirizzario) nella sua memoria. Quando il calcolatore ha finito di copiare, mostrerà il segnale ormai familiare:

READY

Figura 8.4 — Indirizzario del disco TEST/DEMO.

Potete poi ordinare al calcolatore di scrivere sullo schermo l'indirizzario appena caricato mediante il comando:

LIST

e lo schermo si riempirà di una lista come quella raffigurata in figura 8.4.

Un indirizzario può avere più articoli di quanti ne possa contenere lo schermo.

Il Commodore 64 visualizzerà tutta la lista facendo scorrere ("scroll") i primi articoli verso l'alto (e quindi fuori dallo schermo) per far posto all'ultimo articolo scritto. Una volta che questo è stato scritto, il calcolatore segnalerà la sua disponibilità per il comando seguente con la richiesta READY. Potete arrestare l'elencazione in qualsiasi momento premendo il tasto RUN STOP.

Tramite il calcolatore si può effettuare il controllo dell'unità-disco con gli stessi comandi dati per LOAD. Il calcolatore li riconoscerà se seguono il modello: prima il comando, poi il nome del programma tra virgolette, poi una virgola, quindi il numero 8 che apre un collegamento con l'unità connessa al calcolatore. Questi comandi diventano sempre più complicati man mano che aumenta la complessità dei lavori richiesti. Comunque, uno dei programmi contenuti nel disco TEST/DEMO può semplificare i comandi da dare

Pagina mancante

Pagina mancante

INSTRUCTIONS. I comandi /, @ e ↑ sono tutto ciò che vi serve per usare i dischi preregistrati che ubbidiscono ai comandi TEST/DEMO WEDGE.

PROGRAMMI DAL CALCOLATORE AL DISCO

È possibile registrare programmi ed informazioni su ogni disco che abbia un insieme magnetizzato in un formato riconosciuto. Un disco nell'unità che non abbia questo dispositivo è inutile quanto una scrivania vuota per uno scrittore. Se invece è presente, il disco è come un foglio di carta rigata posto sul piano della scrivania e pronto per la scrittura. Potete comandare questa incisione magnetica del disco con un unico comando dal programma WEDGE. Nell'elaborazione, cancellerete ogni precedente informazione sul disco e inserirete un secondo dispositivo in cui memorizzare programmi ed informazioni. Allo stesso tempo si darà al disco un nome. Il sistema calcolatore-disco, quando viene eseguita una scansione, richiede anche un codice di due caratteri per disco che potete fornire all'interno dello stesso semplice comando.

Per preparare un disco a questo modo, dando il nome IL PRIMO e il codice A1, potete inviare il seguente comando:

```
@ N: IL PRIMO,A1
```

oppure

```
OPEN 15,8,15
```

poi

```
PRINT #15,"NO: IL PRIMO, A1"
CLOSE 15,8,15
```

Il calcolatore prenderà il controllo dell'unità-disco e udrete il suono della predisposizione per la registrazione magnetica per qualche attimo. Quando sarà cessato il rumore e si spegnerà la spia rossa dell'unità-disco, il disco sarà stato "inizializzato" e sarà al vostro servizio.

Allo stesso modo di un TEST/DEMO, il disco appena preparato può contenere programmi e file. Potete togliere selettivamente programmi utili da un disco Commodore 64 (come il TEST/DEMO) per porli su un diskette inizializzato.

Per registrare un programma su disco, dovete prima scriverlo sul calcolatore. Ogni programma sarà adatto allo scopo — la citazione da **Amleto** del

cap. 6 o una di vostra creazione —. Potete inserire un programma che volete memorizzare su disco esattamente come fareste prima di farlo girare, o potete caricare un programma da un altro disco o cassetta. Potete, ovviamente, far girare il programma per verificare che il calcolatore faccia esattamente quanto desiderato, e apportare qualunque cambiamento si renda necessario. Quando avrete in memoria il programma desiderato, tutto sarà pronto per il comando che lo preserverà al di fuori del calcolatore.

Quello che vi rimane da fare è dare un nome al programma. Le istruzioni del disco dicono al calcolatore di trattare nomi lunghi fino a 16 caratteri. Se per esempio volete chiamare il vostro programma COLD STORAGE, potete dire al calcolatore di memorizzarlo sotto quel nome mediante il comando:

← COLD STORAGE

oppure

SAVE "COLD STORAGE",8

L'unità-disco produrrà un ronzio e quando questo smetterà e la spia rossa sarà spenta, il programma si troverà in due posti: codificato magneticamente sul disco e, come prima, nella memoria di programma del calcolatore. Non è stato tolto, è stato copiato.

Se scrivete LIST, appariranno sullo schermo le istruzioni di programma. Potete scrivere NEW per cancellare la memoria di programma e tuttavia farlo girare, questa volta tramite disco, mediante il comando:

↑ COLD STORAGE

oppure

LOAD "COLD STORAGE",8

quindi

RUN

Esso sarà allora nuovamente caricato nella memoria del calcolatore. Potrete togliere il disco, spegnere il computer, mandare se volete il disco a fare una gita a Chicago e, al suo ritorno, inserirlo nel drive-box, accendere il calcolatore e scrivere:

↑ COLD STORAGE

oppure

LOAD "COLD STORAGE",8

poi

RUN

e il programma comparirà.

Il calcolatore stava tranquillamente eseguendo gli ordini di un altro disco quando avete comandato ← COLD STORAGE. Ciò pone il nome del programma sull'indirizzario, preceduto da un numero che indica la quantità di spazio che occupa. Se ora mandate un comando:

@$

oppure

LOAD "$",8

poi

LIST

alla fine dell'indirizzario vedrete:

(un numero come 2) COLD STORAGE PRG

Potete cambiare il nome di un programma con un altro comando, mediante il quale il calcolatore lo cercherà sotto il primo nome listato, e lo sostituirà poi con il secondo.

Per esempio, per cambiare il nome da COLD STORAGE a CLANDESTINO, potete inviare il comando:

@ R:CLANDESTINO = COLD STORAGE

oppure

OPEN 15,8,15

poi

PRINT #15,"R0:CLANDESTINO = COLD STORAGE
CLOSE 15,8,15

Così istruito dal comando RENAME, il calcolatore cambia solo il nome del programma, lasciando inalterato il programma stesso e la sua posizione nel CATALOGO. Non c'è nessun bisogno di intasare un disco con un programma che non vi serve più. Per cancellarlo una volta per tutte potete inviare il seguente comando:

@ S:CLANDESTINO

oppure

OPEN 15,8,15

poi

PRINT #15,"S0:CLANDESTINO"

Il calcolatore, su istruzione del disco, farà agire l'unità per trovare il programma, cancellarlo interamente e togliere il suo nome dall'indirizzario.

È possibile spostare individualmente programmi da un disco all'altro usando i comandi che caricano (/), conservano (←) e cancellano (@ S:). Prima caricate il programma da un disco nella memoria del calcolatore, poi lo conservate inviandolo dalla memoria di programma a un altro disco, ed infine lo cancellate dal disco da cui proveniva.

Per spostare un programma come CLANDESTINO da un disco all'altro usando un sistema one-drive dovete per prima cosa inserirlo nella memoria del calcolatore:

/CLANDESTINO

poi togliere il disco dall'unità, inserire un secondo disco, munito della sua incisione magnetica, e inviare il comando:

← CLANDESTINO

quindi reinserire il disco originale e cancellarlo mediante il comando:

@S:CLANDESTINO

Attraverso i comandi conservare, caricare, cancellare e ridenominare potete manipolare e riadattare i programmi su dischi piuttosto liberamente. Ogni disco magnetosensibile può contenere programmi e file d'informazioni allo stesso modo di un disco TEST/DEMO. Potete trasferire selettivamente programmi da un disco Commodore 64 (come TEST/DEMO) a un diskette inizializzato.

Per esempio, si può copiare un programma come FILE SEQUENZIALE tramite il comando

/FILE SEQUENZIALE

oppure

LOAD "FILE SEQUENZIALE",8

Poi, sostituendo il disco TEST/DEMO con uno inizializzato su cui vorreste memorizzare il programma FILE SEQUENZIALE, dare il comando:

← FILE SEQUENZIALE

e il calcolatore duplicherà il programma sul disco che avevate inizializzato.

Usando questa procedura, potete duplicare selettivamente e far muovere programmi sul vostro diskette come meglio ritenete opportuno. In particolare, potete copiare il programma che tratta informazioni e FILE SEQUENZIALE su un disco che sarà usato per scopi particolari, come per esempio contenere file d'informazioni.

USARE PIÙ DI UN'UNITÀ-DISCO

Allo stesso modo in cui diversi dischi vi possono servire come serbatoio di programmi e informazioni, così diverse unità disco aumentano il numero di percorsi di articoli che entrano ed escono dalla memoria del calcolatore. Con due unità, per esempio, potete caricare programmi da, o memorizzarli su, dischi in entrambe le unità.

Per collegare una seconda unità al sistema dovete semplicemente prendere il cavo con la spina a sei poli e inserire un capo nella presa sopra il portafusibili; poi inserire l'altra estremità nella restante presa dell'unità che avete già collegato direttamente al calcolatore. Il collegamento elettrico ora andrà direttamente dal calcolatore a una unità, e da questa alla seconda. Il cavo di alimentazione della seconda unità si collega a un attacco allo stesso modo della prima.

Il calcolatore ora può comunicare con entrambe le unità. Se accendete il computer e una unità, lasciando spenta l'altra, esso eseguirà i comandi sopra descritti facendo lavorare la unità alimentata.

Se accendeste entrambe le unità e deste un comando, come quello di caricare l'indirizzario, il calcolatore sarebbe perplesso. Senza istruzioni su quale delle due unità-disco ricercare l'indirizzario potrebbe scegliere arbitra-

riamente l'una o l'altra, o incepparsi, inviandovi il messaggio che sta caricando l'indirizzario $ quando di fatto non controlla nemmeno un'unità. In questo caso avreste perso il controllo del calcolatore avendolo forzato a prendere decisioni per cui non è stato programmato.

La chiave per evitare questo problema è di rendersi conto che i comandi incorporati dicono al calcolatore che ogni unità è identificata dallo stesso numero. Il numero di questo dispositivo è fissato automaticamente a 8, e lo si può vedere in un comando come LOAD "$",8. Ogni unità collegata al calcolatore, direttamente o tramite un'altra unità, viene automaticamente identificata da questo numero. Ma si può dire al calcolatore di contrassegnare ogni unità con un numero a vostra scelta attraverso comandi che le ricontrassegnino. Questi comandi sono astrusi e sovraccarichi di numeri, e richiedono la creazione di canali di comando e file, anche se non è necessario capire in dettaglio come funzionano per poterli usare. Ecco cosa dovete fare per cambiare il numero con cui il calcolatore chiama un'unità-disco ad esso collegata. Per prima cosa spegnere tutte le unità tranne quella che volete rinumerare.

Inviate la seguente istruzione (ovviamente col tasto RETURN):

```
OPEN 15,8,15
```

Poi mandate questa istruzione se per esempio volete che l'unità-disco abbia il numero 9:

```
PRINT # 15,"M-W"CHR$(119)CHR$(0)CHR$(9+32)CHR$(9+64)
```

Se volete rinumerare un'altra unità, accendetela e ripetete i comandi sopra indicati ponendo un altro numero al posto di 9 nelle ultime due parentesi. Se, per esempio, volete contrassegnare una unità col 22, potete accenderla e seguire questa sequenza:

```
OPEN 15,8,15
PRINT # 15,"M-W"CHR$(119)CHR$(0)CHR$(2)CHR$(22+32)
CHR$(22+64)
```

Il calcolatore riconoscerà il numero di dispositivo che assegnate fino a che non lo cambierete di nuovo o toglierete l'alimentazione.

Poichè i comandi WEDGE presumono un'unità col numero 8, essi entreranno in funzione solo per l'unità a cui è stato automaticamente dato quel numero o per una a cui tale numero è stato assegnato. Se per esempio i numeri 8 e 9 sono assegnati a due unità, potete usare la forma estesa per controllare ognuna delle unità, e i comandi WEDGE per controllare la unità 8.

Per esempio potrete copiare un programma da un disco di unità 8 su un altro in unità 9 con una sequenza di comandi come la seguente. Per prima cosa scrivete:

LOAD "FILE SEQUENZIALE",8

Poi attendete la richiesta READY e scrivete:

SAVE "FILE SEQUENZIALE",9

FATTI E FILE SU DISCO

I programmi sono abbastanza utili, ma diciamo la verità, quello che realmente si vuole dal calcolatore è trattare informazioni-dati, fatti, numeri, nomi, indirizzi, tempi, località, quantità, descrizioni.

Qualunque informazione che possa essere scritta può anche essere posta su disco. È possibile memorizzarla sotto un determinato nome nell'indirizzario ed evidenziarla sullo schermo tramite il calcolatore. Potete ordinare al calcolatore di lavorare su di essa, elaborandola secondo le istruzioni di programma.

Come schedari in un archivio, l'informazione su un disco è a vostra disposizione per la lettura o per essere usata nei programmi al posto di istruzione DATA e INPUT. Comunque, il calcolatore ha bisogno di istruzioni più dettagliate per porre su disco un'informazione che non sia organizzata come programma, e per trattenerla poi in memoria.

Uno dei programmi sul disco TEST/DEMO tratta informazioni su disco ed ha due parti operative. Una parte ordina al calcolatore di porre informazioni di tastiera su un disco inizializzato, l'altra gli dice di leggere le informazioni dal disco. Programmi più elaborati, che portano il calcolatore a gestioni complesse di informazioni sono disponibili sul mercato su dischi pre-registrati. Se siete portati, potete scriverne voi stessi creando dal programma del disco TEST-DEMO. Esso infatti fornisce una serie di comandi che ordinano al calcolatore trasferimenti di informazioni.

Quando trasporta informazioni — chiamate anche **text files** — da e verso un disco, il Commodore 64 le tratta allo stesso modo di stringhe di parole in un programma. Se ordinate numeri, parole, comandi o dichiarazioni di sentimenti in questi text file, il calcolatore li tratterà come semplici gruppi di caratteri.

Potete usare il programma senza studiare il funzionamento. È anche possibile listare il programma lasciando solo le istruzioni di gestione di file. Que-

sta sequenza può poi venir aggiunta ad altri programmi per elaborare informazioni e, se volete, per creare nuovi file.

Il programma che ordina al calcolatore di creare informazioni è listato sul disco TEST/DEMO come FILE SEQUENZIALE. Potete usarlo più facilmente se prima lo copiate su un disco che avrete "inizializzato" per mezzo di un formato magnetico riconosciuto inciso. Quanto poi comanderete:

↑ FILE SEQUENZIALE

oppure

LOAD "FILE SEQUENZIALE", 8
RUN

il calcolatore pulirà lo schermo e scriverà le descrizioni e le indicazioni per l'uso del programma. A questo punto potete ignorare le prime due righe e seguire le indicazioni IMMETTERE UNA PAROLA, VIRGOLA, NUMERO. Se volete immettere, come primo della lista, il nome AMBROGIO e il numero 1, allora potrete scrivere dopo la richiesta A$,B:

AMBROGIO,1

inviandolo poi al calcolatore premendo il tasto RETURN. Poi potete inviare un altro nome e numero:

LUCA,357

e un terzo nome e numero:

MARCO,2

Se volete che il vostro elenco contenga solo questi tre articoli, potete segnalare al calcolatore di chiuderlo mediante la risposta:

END

inviata premendo RETURN. Appariranno due punti di domanda come richiesta. Se schiacciate ancora RETURN il calcolatore, che agisce ai comandi del programma FILE SEQUENZIALI, lo considererà come indicazione che il file è terminato. Seguendo gli ordini della seconda parte di questo programma, il Commodore 64 scriverà la riga READ SEQ TEST FILE e a questo punto SEQ TEST FILE sarà il nome che il programma ha dato all'elenco di articoli da voi scritti. Infine il calcolatore scriverà sullo schermo l'elenco di articoli e numeri.

Dopo aver fatto girare il FILE SEQUENZIALE vedrete che il computer ha aggiunto il file alla fine dell'indirizzario sotto il nome SEQ TEST FILE e l'elenco apparirà così:

```
1 "SEQ TEST FILE" SEQ
```

Le informazioni sono state copiate articolo per articolo dal disco al calcolatore, come se fossero state battute alla tastiera dopo un'istruzione INPUT, pur essendo sempre presenti sul disco.

CAPITOLO 9

FARE E RIFARE I PROGRAMMI

Un programma può avere un'esistenza alquanto incerta. Lo si può far funzionare ripetutamente, svolgendo ogni volta la stessa funzione, e può essere modificato per adattarsi a necessità diverse. Per usare il calcolatore in modo intelligente e creativo non bisogna scrivere un programma nuovo per ogni progetto, ma piuttosto cercare di creare e adattare programmi o parti di programma che abbiano dimostrato la loro utilità. Di fatto è molto più facile partire con gruppi di comandi i cui risultati siano già noti piuttosto che porsi davanti a uno schermo vuoto ad ogni nuovo progetto!

AGGIUNGERE PARTI DI PROGRAMMA

È possibile dire al Commodore 64 di unire assieme in memoria programmi (o sue parti) da voi considerati particolarmente interessanti o utili pur non sapendo molto riguardo ad essi. Per creare il vostro stile di programmazione potete trovare sequenze di istruzioni già pronte e di uso frequente. Usando questo metodo, è possibile sfruttare quelle sequenze che vi sono familiari e porle nel programma utilizzandole in ogni sua parte.

Per esempio, per vedere un nome in basso, al centro dello schermo, potete creare questo programma di due righe:

```
NEW
  10 DATA BONAPARTE L'AMBIZIOSO
1000 READ A$: PRINT "(premere SHIFT e CLR/HOME)" A$
```

Potete utilizzare di nuovo il dispositivo di cancellazione dello schermo, posizionamento e stampa di riga 1000, ordinando al calcolatore di svolgere i comandi di questa riga per altri nomi, (cioè altri valori di A$). Può essere ag-

giunta al programma la coppia di comandi GOSUB e RETURN che manda il calcolatore a riga 1000 e ritorno mediante queste istruzioni:

```
   20 GOSUB 1000
 1010 RETURN
```

Una terza istruzione, END, impedirà al calcolatore di agire di nuovo su riga 1000. Essa può essere posta ovunque prima di riga 1000, così:

```
  100 END
```

Incontrando il comando GOSUB il calcolatore salta ogni riga intermedia ed agisce immediatamente su riga 1000. Poi, in sequenza normale, esegue i comandi delle righe che vengono dopo 1000. Quando il calcolatore incontra il comando RETURN, viene mandato indietro all'istruzione GOSUB e alla riga immediatamente seguente. Avendo eseguito i comandi tra riga 1000 e l'istruzione RETURN, il calcolatore riprende la sequenza a riga 100 che gli dice di fermarsi.

Il comando GOSUB può essere così tradotto: "Fissare il numero di riga di questa istruzione in memoria, poi eseguire i comandi partendo dal numero di riga indicato". Il comando RETURN si traduce: "Ritornare alla marcatura del comando GOSUB e continuare con il numero di riga che segue".

Usando questi due comandi potrete inviare il calcolatore a un gruppo di istruzioni in qualunque parte del programma. Il comando GOSUB può essere usato diverse volte per indirizzare il calcolatore a un gruppo di istruzioni, come nella sequenza che segue:

```
   10 DATA BONAPARTE L'AMBIZIOSO
   14 GOSUB 1000
   20 DATA ALBERTO IL SOTTOVALUTATO
   24 GOSUB 1000
   30 DATA MELVILLE IL MAGNIFICO
   34 GOSUB 1000
  100 END
```

Ovunque compaia, il comando GOSUB 1000 ordina al calcolatore di operare su riga 1000 e seguenti. L'istruzione RETURN lo rimanda alla sequenza principale di istruzioni DATA e GOSUB. Se aggiungete un'iterazione vuota a riga 1000, per ritardare i cambiamenti del display:

```
 1000 FOR T = 1 TO 500 : NEXT T : READ A$ : PRINT
      "(premere SHIFT e CLR/HOME)" A$
```

e poi fate svolgere il programma, vedrete il nome assegnato a ogni istruzione DATA scritto nell'angolo in alto a sinistra dello schermo, uno dopo l'altro.

Potete dire al calcolatore di agire su qualsiasi numero di gruppi di comandi — dette **subroutine** — seguendo qualsivoglia sequenza. Con un altro gruppo di comandi è possibile aggiungere una funzione grafica al programma:

```
2000 N$ = "(premere SHIFT e W)"
2010 FOR I = 1 TO 6
2020 READ A(I)
2030 PRINT "(premere CLR/HOME)"
2040 Y = 23 - A(I):X = 5 * I
2050 FOR V = 1 TO Y : PRINT "(premere CURSOR UP/DOWN)"; :
     NEXT V
2060 PRINT TAB (X) N$
2070 NEXT I
2080 PRINT "(premere CLR/HOME)" : RETURN
```

Questo gruppo di comandi indirizzerà il calcolatore ai valori READ da un'istruzione DATA e poi, con una piccola manipolazione aritmetica dei valori di X e Y, lo porterà a rappresentare graficamente quei valori.

Ora potete utilizzare il comando GOSUB in altri modi. Espandendo le istruzioni DATA di righe 10, 20 e 30, è possibile far lavorare i comandi che iniziano in riga 2000 sui valori:

```
10 DATA BONAPARTE L'AMBIZIOSO,5.5,7,11,17,9,1
20 DATA ALBERTO IL SOTTOVALUTATO,2,3,16,17,18,19
30 DATA MELVILLE IL MAGNIFICO,5,10,7,15,17,12
```

Questi valori potrebbero per esempio rappresentare pesi annui, o produttività, o dimensione del vocabolario.

Per indirizzare il calcolatore alla subroutine grafica dovete aggiungere, dopo ognuno dei comandi GOSUB 1000, un GOSUB 2000:

```
18 GOSUB 2000
28 GOSUB 2000
38 GOSUB 2000
```

Girando, questo programma dirà al calcolatore di scrivere prima il nome e poi rappresentare graficamente la serie corrispondente di numeri.

Poichè il comando GOSUB manda il calcolatore a un altro numero di riga, esso può essere usato in una subroutine per mandare il calcolatore a un secondo gruppo di comandi passando attraverso il primo.

Incontrando il comando RETURN del secondo gruppo, esso sarà rimandato al primo e, incontrando il comando RETURN del primo, sarà rimandato alla sequenza principale.

Usando a questo modo la coppia GOSUB-RETURN potete aggiungere sottigliezze ai programmi in modo "modulare". È possibile per esempio migliorare l'aspetto grafico di questo programma aggiungendo righe quadrettate con le quali confrontare ogni rappresentazione. Per ottenere ciò si dovrebbe raggruppare una sequenza di comandi come due subroutine, una che traccia linee verticali che partono da riga 3000:

```
3000 PRINT "(premere CLR/HOME)" (premere CURSOR
     UP/DOWN)"
3010 FOR V = 1 TO 20
3020 FOR I = 0 TO 39 STEP 5
3030 PRINT TAB (I) "(premere C= e G) (premere SHIFT e CURSOR
     UP/DOWN) (premere SHIFT e CURSOR UP/DOWN) (premere
     SHIFT e CURSOR UP/DOWN)"
3040 NEXT I
3050 PRINT
3060 NEXT V
3070 RETURN
```

e un'altra che traccia linee orizzontali con numeri in scala a partire da riga 4000:

```
4000 PRINT "(premere CLR/HOME) (premere CURSOR UP/DOWN)
     (premere CURSOR UP/DOWN)"
4010 FOR Y = 0 TO 20 STEP 5
4020 FOR X = 3 TO 35
4030 PRINT 20 - Y TAB (X) "(premere C= e@) (premere SHIFT e
     CURSOR UP/DOWN)"
4040 NEXT X
4050 IF Y = 20 THEN 4070
4060 FOR S = 1 TO 5:PRINT:NEXT S
4070 NEXT Y
4080 RETURN
```

È possibile ordinare al calcolatore di disegnare il reticolo subito dopo aver letto i valori dagli ordini della subroutine che inizia in riga 1000 aggiungendo questa istruzione:

```
1007 GOSUB 3000 : GOSUB 4000
```

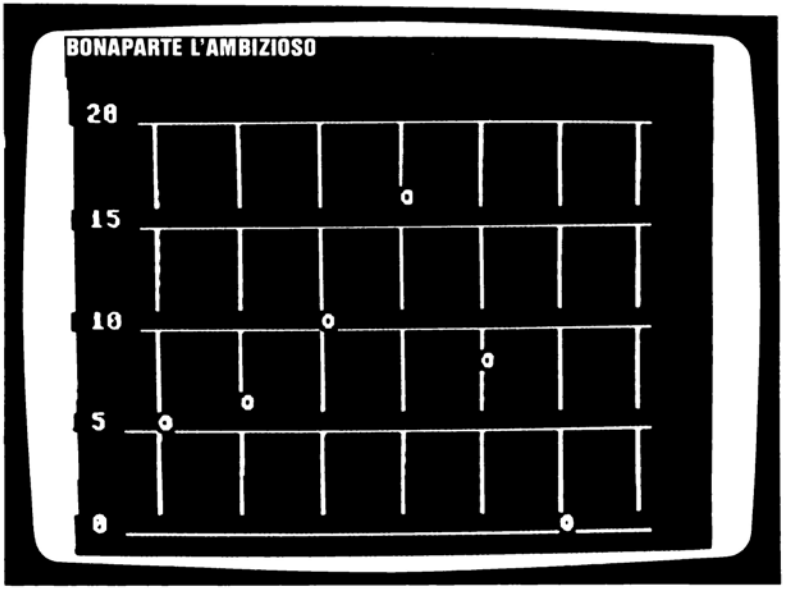

Figura 9.1 — Un reticolo creato da quattro subroutine.

Allora per prima cosa lo schermo si pulirà e in alto apparirà il nome "BONAPARTE L'AMBIZIOSO". Quindi il calcolatore traccerà le righe del reticolo e poi scriverà i valori dalla prima istruzione DATA, come illustrato in figura 9.1. Poi lo schermo si pulirà ancora e inizierà l'elaborazione dei valori dell'istruzione DATA successiva, continuando così fino a trattare ogni serie di valori delle tre istruzioni DATA fermandosi all'ultimo display. Il calcolatore agisce sui comandi GOSUB e RETURN in coppia. Se incontra un'istruzione RETURN senza aver prima trovato GOSUB, esso seguirà i comandi incorporati segnalandovi un errore di messaggio

RETURN SENZA GOSUB ERROR IN (numero di riga)

Potete evitare ciò ponendo un'istruzione END o STOP dopo la sequenza principale, ma prima della prima istruzione di subroutine. In questo caso, abbiamo posto un END a riga 100. Allo stesso modo del comando GOTO, GOSUB invia il calcolatore al numero di riga listato nel comando. È possibile usare una variante dei comandi GOSUB-RETURN per mandare il calcolatore, in modo condizionale, dentro e fuori dalla sequenza principale di istruzioni. Come un comando ON-GOTO ordina al calcolatore di abbandonare la normale sequenza a certe condizioni numeriche, così un comando ON-GO-

SUB indirizza il calcolatore a particolari subroutine. Potete usare questo comando per indirizzare il calcolatore a uno dei diversi gruppi di comandi, da cui più tardi sarà rimandato indietro per riprendere di nuovo la sequenza principale.

L'istruzione ON-GOSUB può essere per esempio usata nel programma di rappresentazione grafica per disegnare selettivamente parti del reticolo contro cui sarà tracciato il grafico delle informazioni. Un gruppo di comandi, come quelli che iniziano in riga 3000, possono far disegnare al calcolatore righe verticali mentre un altro gruppo, come quello con inizio a riga 4000, può far tracciare righe orizzontali.

Modificando riga 1007 in un'istruzione condizionale potete, (a seconda della scelta fatta durante ogni svolgimento di programma) sia usare, sia saltare le subroutine di 3000 o 4000, o una subroutine nuova in riga 5000. Questa crea l'intero reticolo, mandato il calcolatore a tracciare righe prima verticali, poi orizzontali. L'intero reticolo sarà tracciato con queste istruzioni:

```
5000 GOSUB 4000
5010 GOSUB 3000
5020 RETURN
```

Il comando che indirizza il calcolatore ad ognuna di quelle subroutine, per determinati valori della variabile D, è:

```
1007 ON D GOSUB 3000,4000,5000
```

Questo indirizza il calcolatore allo stesso modo del comando ON-GOTO, confrontando il valore della variabile con le gamme di valori del comando ON-GOSUB: uno o più, ma meno di due, e così via. Potete far richiedere al calcolatore il valore di D aggiungendo una richiesta e un'istruzione INPUT, come queste:

```
1005 PRINT "PREMERE IL NUMERO DI DISPLAY CHE VOLETE:"
1006 INPUT "1-VERTICALE; 2-ORIZZONTALE; 3-RETICOLO
     COMPLETO"; D
```

Mentre il programma gira, vedrete la richiesta INPUT in cima allo schermo. Se effettuate una scelta nella gamma 1, 2 o 3, il calcolatore per prima cosa traccerà le linee a seconda della scelta, poi rappresenterà graficamente i valori dell'istruzione DATA. Se scrivete una scelta al di fuori della gamma, il calcolatore andrà semplicemente alla riga successiva e non verrà disegnata nessuna riga; i valori DATA saranno rappresentati su uno schermo privo di righe.

Potete creare una biblioteca di utili subroutine dando loro numeri di riga molto alti per evitare interferenze con una sequenza principale dai numeri più bassi. Infatti il vostro programma principale qualche volta può essere niente di più di una serie di richieste di subroutine con qualche eventuale sequenza di INPUT o DATA. Nel comporre tale programma gli unici elementi che devono corrispondere sono le variabili, comuni alle subroutine e che ivi agiscono. Potete essere sicuri di questo riscontro controllando che il nome di una data variabile rappresenti la stessa quantità ovunque venga usata. La sequenza principale di tale programma potrebbe essere semplice come:

```
10 GOSUB 1000
20 GOSUB 2000
30 GOSUB 3000


100 END
```

L'istruzione finale END impedisce al calcolatore di andare alle subroutine successive. Seguendo questa struttura potrete risparmiarvi del lavoro, avvantaggiandovi delle subroutine già scritte da voi o da qualcun altro, dandovi un maggior controllo sul vostro calcolatore. Vi potrete concentrare nella combinazione di elementi già esistenti piuttosto che nella scrittura di comandi dettagliati.

RICOSTRUZIONE DI PROGRAMMI PRESI A PRESTITO

Non tutti i programmi vengono creati con il processo faticoso di aggiungere un comando dietro l'altro. Potete prendere a prestito programmi esistenti, alterarli per adattarli alle vostre necessità sfruttando così gli sforzi fatti per il programma originale. Utilizzando un programma su disco — per esempio il FILE SEQUENZIALE del disco TEST/DEMO — potete usare le sue capacità in programmi vostri.

Il programma FILE SEQUENZIALE, che svolge due compiti distinti di gestione di file, ponendo file d'informazioni su disco e leggendoli, può essere diviso in due parti separate. Molte delle sue istruzioni non fanno altro che descrivere le funzioni del programma, e possono essere tolte. Eliminando questi comandi che non vi servono, potete ripulirlo un po' dagli eccessi dell'originale.

```
LIST -95

1  REM **************
2  REM *     ESEMPIO
3  REM * LEGGERE E SCRIVERE
4  REM * UN FILE DI DATI
5  REM * SEQUENZIALE
6  REM ***************
10 PRINT"              INIZIALIZZARE IL DISCO"
20 DIMA$(25)
30 DIMB(25)
40 OPEN15,8,15,"10"
60 GOSUB 1000
70 CR$=CHR$(13)
80 PRINT
90 PRINT" ↳ SCRIVERE SEQ TEST FILE"
95 PRINT

READY.
```

Figura 9.2 — Le istruzioni del programma FILE SEQUENZIALE che predispongono il calcolatore alla gestione di un file.

Non è necessario comprendere ogni dettaglio di un programma per poterlo usare in tutte le sue parti. Per esempio, se guardate l'intero elenco del FILE SEQUENZIALE, il contenuto di uno schermo alla volta, potete vedere che il programma è diviso in quattro parti, ognuna separata dalle note di un gruppo di istruzioni REM. Il primo gruppo di istruzioni REM, numerate da 1 a 6, descrive la funzione del programma che segue, ESEMPIO LEGGERE & SCRIVERE UN SEQ DATA FILE. Le istruzioni che seguono quel gruppo di note, righe da 10 a 95 (vedi figura 9.2) contengono comandi comuni ad entrambe le funzioni del programma.

Il gruppo successivo, da 100 a 105, descrive piuttosto chiaramente la parte di programma che comanda la formazione di un file, SCRIVERE SEQ TEST FILE. Queste istruzioni occupano le righe da 110 a 160 (figura 9.3).

Le note da 200 a 205 descrivono le istruzioni, righe da 206 a 420 che ordinano la lettura nel calcolatore di informazioni dal file, SEQ TEST FILE, prodotte dalla prima metà del programma, LEGGERE SEQ TEST FILE (figura 9.4).

Infine, le note in righe da 1000 a 1006 descrivono una sequenza di comandi di contingenza in una subroutine, istruzioni da 1010 a 1060, che mo-

```
LIST 100-160

100 REM *************
101 REM *
102 REM *    SCRIVERE SEQ
103 REM *    TEST FILE
104 REM *
105 REM *************
110 OPEN2, 8, 2, "CO : SEQ TEST FILE , S, W"
115 GOSUB 1000
117 PRINT "IMMETTERE UNA PAROLA, VIRGOLA, NUMERO"
118 PRINT "IMMETTERE LA PAROLA 'END' PER FERMARE"
120 INPUT "A$, 8"; A$, 8
130 IF A$ = "END" THEN 160
140 PRINT # 2, A$", "STR$(B)CR$;
145 GOSUB 1000
150 GOTO 120
160 CLOSE 2

READY.
```

Figura 9.3 — Le istruzioni del programma FILE SEQUENZIALE che inviano informazioni a un file su disco.

```
LIST 200-300
200 REM *************
201 REM *
202 REM *    SCRIVERE SEQ
203 REM *    TEST FILE
204 REM *
205 REM *************
206 PRINT
207 PRINT" LEGGERE SEQ TEST FILE"
208 PRINT
210 OPEN2, 8, 2, "0 : SEQ TEST FILE , S, R"
215 GOSUB 1000
220 INPUT # 2, A$(1), B(1)
224 RS = ST
225 GOSUB 1000
230 PRINTA$(1), B(1)
240 IFR S = 64 THEN 300
250 IF RS < >0 THEN 400
260 I = I + 1
270 GOTO 220
300 CLOSE 2
READY.
```

Figura 9.4 — Le istruzioni del programma FILE SEQUENZIALE che prelevano informazioni da un file su disco. (segue)

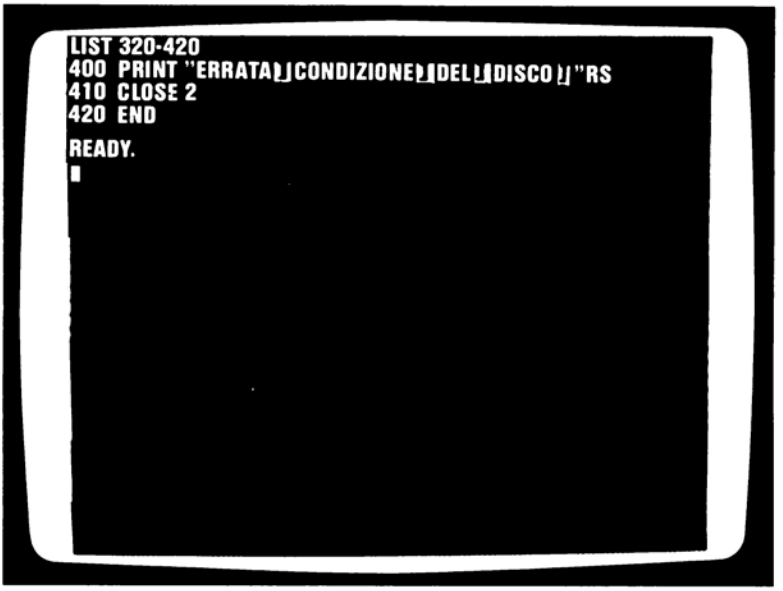

Figura 9.4 — Le istruzioni del programma FILE SEQUENZIALE che prelevano informazioni da un file su disco.

streranno un codice per quei problemi che possono presentarsi per comandi non riconosciuti tra il calcolatore e l'unità-disco, LEGGERE IL CANALE DI ERRORE (figura 9.5).

Per eliminare quella parte del programma che crea il file nella prima posizione, avete bisogno solo delle righe fino a 160, e della subroutine da riga 1010 in poi. Potete iniziare a modellare questo programma per le vostre necessità eliminando le righe da 200 a 410. Fate ciò, ricordate, mandando ogni numero di riga con il tasto RETURN:

200
201

.

400
410

Il programma che rimane nella memoria del calcolatore vi richiederà di formare un file non superiore a 25 combinazioni di numeri-parole, che chiamerà SEQ TEST FILE. Si tratta di un programma che ha diverse istruzioni in

```
LIST 1000-1060

1000 REM ************
1001 REM *
1002 REM *  LEGGERE
1003 REM * IL CANALE
1004 REM * D'ERRORE
1005 REM *
1006 REM ************
1010 INPUT # 5, EN, EMS, ET, ES
1020 IF EN = 0 THEN RETURN
1030 PRINT"              ERRORE SUL DISCO"
1040 PRINTEN; EMS; ET; ES
1050 CLOSE 2
1060 END
READY.
```

Figura 9.5 — Le istruzioni del programma FILE SEQUENZIALE che agiscono su un problema.

```
LIST -140

20 DIMA$(25)
30 DIMB(25)
40 OPEN15, 8, 15, "10"
60 GOSUB 1000
70 CR$ = CHR$(13)
80 PRINT
90 INPUT "NOME DEL FILE DA CREARE"; FS
110 OPEN 2, 8, 2, "00 : "+F$+", S, W"
115 GOSUB 1000
117 PRINT "IMMETTERE UNA PAROLA, VIRGOLA, NUMERO"
118 PRINT "IMMETTERE LA PAROLA 'END' PER FERMARE"
120 INPUT "A$, 8"; A$, 8
130 IF A$ = "END" THEN 160
140 PRINT # 2, A$", "STR$ (B)CR$;
READY.
```

Figura 9.6 — Un programma di scrittura derivato dal programma FILE SEQUENZIALE. (segue)

più rispetto a quelle che realmente comandano detta funzione. Potete ripulirlo da queste istruzioni superflue eliminando le note delle righe da 1 a 6, l'istruzione ridondante PRINT di riga 10, nonchè le note e la spaziatura delle righe da 95 a 105. (Ricordatevi, comunque, che voi o un altro utente potrebbero apprezzare in seguito la possibilità di avere delle righe REM riservate). Il programma che rimane dopo questa pulizia dirà al calcolatore di formare un file tramite la vostra interazione. Comunque se lo fate girare più di una volta per creare file differenti, esso darà il medesimo nome ad ogni serie di dati, SEQ TEST FILE. Alterando due istruzioni è possibile modificare il programma per assegnare un nome di vostra scelta ad ogni file che intendete creare.

L'istruzione che dà il nome al file da porre su disco è a riga 110. La potete rendere più generica cambiando semplicemente la parte che collega il file a una variabile. Per fare ciò occorre il tipo di aggiunta che si può effettuare sulle stringhe di parole con il segno più. Usando la variabile F$ per indicare il nome del file, potete sostituire riga 110 con l'istruzione:

```
110 OPEN 2,8,2 " @ 0:"+F$ + ",S,W"
```

Ora il programma dice al calcolatore di dare ad ogni file il nome assegnato alla variabile F$. Potete fare questa assegnazione quando il programma è svolto, includendo un'istruzione che vi chieda un nome di file. Lo farà un'istruzione INPUT al posto della PRINT di riga 90:

```
90 INPUT "NOME DEL FILE DA CREARE";F$
```

Il calcolatore contiene ora un programma versatile che può creare file dopo file in risposta alle vostre immissioni e può dare a ognuna un nome diverso. Usando questo programma potete creare molti file, ognuno elencato per nome nell'indirizzario del disco su cui è memorizzato.

Il programma che ne risulta e che vedrete mandando il comando LIST (come illustrato in figura 9.6), è pulito e a punto.

Ordinerà al calcolatore di chiedervi il nome del file che desiderate produrre e poi, attraverso le richieste della sua istruzione INPUT, radunerà e invierà all'unità disco gli elementi da voi forniti dalla tastiera.

Potete conservare questo programma così com'è, ponendolo su un disco su cui intendete memorizzare i file sotto il nome CREARE FILE:

```
← CREARE FILE
```

oppure

```
SAVE "CREARE FILE",8
```

```
LIST 150-

150 GOTO 120
160 CLOSE 2
170 END
1000 REM ************
1010 INPUT # 5, EN, EM$, ET, ES
1020 IF EN = 0 THEN RETURN
1030 PRINT"        ERRORE SUL DISCO"
1040 PRINT EN; EM$; ET; ES
1050 CLOSE 2
1060 END
READY.
```

Figura 9.6 — Un programma di scrittura derivato dal programma FILE SEQUENZIALE.

D'altra parte, volendo potete includere questo programma in uno più grande o apportare ulteriori aggiustamenti.

Come già detto, il programma di gestione file riserva spazio solo per 26 immissioni. Sono le righe 20 e 30 che fanno l'assegnazione:

```
20 DIM A$(25)
30 DIM B(25)
```

Potete aumentare il numero di elementi da porre nel file che create modificando quelle istruzioni per riservare più spazio nella memoria del calcolatore. Se volete avere spazio per 100 articoli, per esempio, semplicemente inviate nuove istruzioni a questo scopo:

```
20 DIM A$(100)
30 DIM B(100)
```

Allo stesso modo, si può modificare il formato degli articoli inseriti cambiando l'istruzione INPUT di riga 120 di modo che accetti qualunque numero di articoli. Con lo stesso attento lavoro potete creare un programma di lettura da un'altra parte del programma FILE SEQUENZIALE. In questo caso,

caricate il programma nella memoria del calcolatore e iniziate a ripulire. Di nuovo, si possono eliminare le righe da 1 a 10:

```
1
2
.
10
```

Come pure si possono togliere le righe da 70 a 206, che comandano la formazione del file:

```
70
80
.
206
```

Sono inutili anche le istruzioni da 1001 a 1006, perchè si tratta solo di note:

```
1001
1002
1003
1004
1005
1006
```

Così com'è ora, il programma dirà al calcolatore di trovare e leggere in memoria gli articoli da un file su disco chiamato SEQ TEST FILE. Cambiando due istruzioni, potete modellare il programma in modo da chiedere il nome del file, e poi trovarlo per nome. La riga che dice al calcolatore di cercare SEQ TEST FILE, la 210, può essere modificata per dirgli di cercare un file con qualsiasi nome, memorizzato sotto la variabile F$, così:

```
210 OPEN 2,8,2,"0" +F$+",S,R"
```

È possibile dire al calcolatore di chiedervi il nome di un file mediante l'istruzione:

```
207 INPUT "NOME DEL FILE DA RICERCARE";F$
```

Il programma ora in memoria (figura 9.7) cercherà un disco con il file da

```
LIST -240

20 DIM A$(100)
30 DIM B(100)
40 OPEN 15, 8, 15, "10"
60 GOSUB 1000
207 INPUT "NOME DEL FILE DA RICERCARE"; FS
210 OPEN 2, 8, 2, "0 : "+ F$ +", S, R"
215 GOSUB 1000
220 INPUT # 2, A$(1), B(1)
224 RS = ST
225 GOSUB 1000
230 PRINT A$(1), B(1)
240 IFR S = 64 THEN 300

READY.
```

Figura 9.7 — Un programma di lettura derivato dal programma FILE SEQUENZIALE. (segue)

```
LIST 250-

250 IF RS<>0 THEN 400
260 I = I + 1
270 GOTO 220
300 CLOSE 2
310 END
400 PRINT "ERRATA CONDIZIONE DEL DISCO" RS
410 CLOSE 2
420 END
1000 REM ************
1010 INPUT # 5, EN, EM$, ET, ES
1020 IF EN = 0 THEN RETURN
1030 PRINT"    ERRORE SUL DISCO"
1040 PRINTEN; EM$; ET; ES
1050 CLOSE 2
1060 END
READY.
```

Figura 9.7 — Un programma di lettura derivato dal programma FILE SEQUENZIALE.

voi indicato in risposta alla domanda del calcolatore, e poi stamperà ogni elemento del file come richiesto da riga 230:

```
PRINT A$(I),B(I)
```

È possibile usare questo programma per ricercare e listare file, oppure alterarlo ulteriormente per usare questi elementi in altro modo. Potete aggiungere qualsiasi istruzione vogliate, fintanto che queste tengono in considerazione la forma delle variabili nelle istruzioni INPUT e sue collegate. Al contrario, è possibile cambiare queste variabili per farle eventualmente meglio aderire al vostro programma.

Questo approccio può essere usato ogni volta che vogliate far operare il calcolatore sulle informazioni che riceve da un file. Eliminando le istruzioni che trovate superflue, potete rendere fluido e adattabile qualsiasi programma pre-registrato listato dal calcolatore.

CAPITOLO 10

LA STORIA NASCOSTA

Per chi ha preso confidenza con l'uso di programmi e comandi, il calcolatore non è più quello strano congegno che era sembrato all'inizio. Anche se imparando ad adoperare il Commodore 64 avete ottenuto una conoscenza della logica che sta dietro la macchina, molte delle sue possibilità restano sepolte nella sua memoria e nella sua concezione. C'è però il modo di scoprire questi segreti; potete addentrarvi nel calcolatore senza trovarvi sommersi da microchip e grovigli di fili. Più comandi vi condurranno nel labirinto dei codici dei numeri che comandano il calcolatore. Un assortimento di circuiti extra faranno del Commodore 64 qualcosa di più della macchina che era, e uno sguardo all'interno rivelerà un personal computer inimmaginabile solo pochi anni fa.

UN PASSO OLTRE L'ASSEMBLATORE

L'**assemblatore** è come un ponte tra voi e l'**elaboratore,** che controlla il flusso di informazioni attraverso i circuiti del calcolatore. Questi circuiti sono incorporati nei microchip, come illustrato in figura 10.1. Nel calcolatore l'elaboratore e i serbatoi di informazioni (i "chip di memoria") sono alquanto separati l'uno dagli altri. La memoria del calcolatore è stata ideata in modo tale che ogni informazione sia memorizzata per numero, il quale rappresenta la posizione di quel dato. Anche se un elaboratore può cambiare le informazioni in una posizione della memoria, questa posizione numerata continuerà ad esistere, senza considerare il valore in essa contenuto, come un indirizzo fisso con inquilini diversi.

I comandi PEEK e POKE

Anche se si aspetta comandi nel linguaggio BASIC, il calcolatore può essere mandato a indirizzi interni. È possibile modificare le risposte automati-

che del calcolatore cambiando i valori numerici in queste posizioni. Un comando manderà i valori nelle posizioni di memoria, aggirando i soliti processi di conversione.

Sviluppare una particolare capacità sul linguaggio numerico del calcolatore richiede uno studio noioso e pesante, cui si sono dedicati altri libri. A meno che non abbiate la passione di dare al vostro calcolatore istruzioni che vanno al di là dei comandi del BASIC, si tratta di uno studio che non raccomandiamo.

Il linguaggio numerico del calcolatore è espresso anche in termini di codici di carattere che consistono in numeri e lettere. Questi codici possono rappresentare ogni possibile valore e posizione nella memoria del Commodore 64. Con questo linguaggio numerico è possibile inviare il calcolatore a una posizione particolare di memoria, per esempio A65E, dove una serie di comandi SHIFT-HOME dice al calcolatore di ripulire lo schermo e posizionare il cursore.

I programmi di queste istruzioni codificate possono formare liste apparentemente senza fine che offuscano la visione e indeboliscono l'ingegno di chiunque, tranne forse i più decisi ed entusiasti. Questo metodo di notazione è conosciuto come **esadecimale.**

Per fortuna, esiste un modo più comodo. Nel BASIC esiste un comando che usa i più comuni numeri decimali. Potete usarlo per modificare le istruzioni interne del calcolatore in altri ordini da voi specificati. Questo comando è POKE che invia il calcolatore a una posizione numerata dove viene cambiato il valore numerico che avete comandato.

Per rendervi conto di quanto possa essere utile il comando, considerate l'esempio di modificare il display dello schermo. Il Commodore 64 scrive automaticamente i caratteri in azzurro su fondo blu con contorno azzurro.

Questo display è dato (ogni volta che date alimentazione) da valori copiati nella memoria attiva da parte della memoria interna non modificabile. Variando i valori in queste posizioni di memoria attiva potete cambiare il modello di scrittura sullo schermo. Una posizione stabilisce il colore della "finestra", un'altra il colore del bordo.

Il calcolatore segue le sue istruzioni interne stabilendo i colori dello sfondo e del bordo. Cambiando queste istruzioni, è possibile formare display di ogni combinazione di colori. Per fare ciò, potete usare il comando POKE, che passa oltre l'assemblatore per parlare più direttamente con l'elaboratore. È possibile ordinare all'elaboratore di stabilire un nuovo colore per lo sfondo dicendogli di porre un altro valore nella posizione di memoria 53281, dove è automaticamente memorizzato un valore per uno schermo di colore blu.

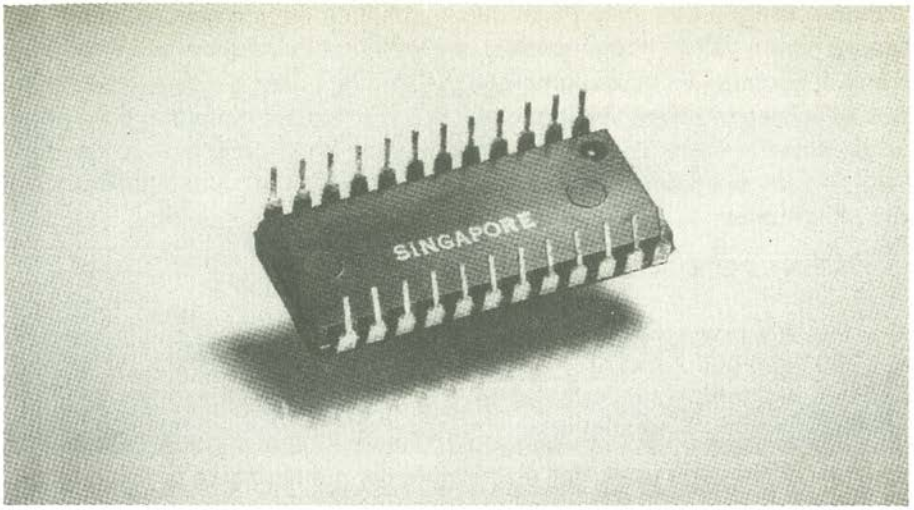

Figura 10.1 — Un microchip del tipo usato nei microelaboratori.

Se volete che il colore sia nero, numero di codice 0, potete usare il comando:

POKE 53281,0

Questo comando POKE dice: "Trovare il numero di posizione in memoria 53281 e memorizzare il valore 0 a quell'indirizzo". Allo stesso modo potete ordinare all'elaboratore di portare a nero il colore del bordo del display. L'elaboratore cerca le istruzioni per questa parte del display al numero di indirizzo di memoria 53280. Potete usare il comando:

POKE 53280,0

In combinazione, questi due comandi e i valori al loro interno, (sfondo colore 0 e bordo colore 0), dicono al calcolatore di rappresentare uno sfondo nero per l'intero schermo. Essi vi lasciano effettivamente prendere il controllo delle istruzioni automatiche incorporate portando il calcolatore a un modo di operazione calibrato per le vostre effettive necessità. Ognuna delle 65536 posizioni di memoria del Commodore 64 sarà resa disponibile dal comando POKE per i vostri nuovi valori. Quale strumento per personalizzare e riprogrammare la prestazione del calcolatore, il comando POKE vi dà l'assoluto controllo ordinando al calcolatore di effettuare le modifiche prima di passare alla riga successiva.

Potete usare il comando PEEK, di cui abbiamo parlato nel capitolo 6, per considerare il valore in ogni posizione di memoria. Esso dirà al calcolatore prima di cercare l'indirizzo numerato che gli avete dato, come fa il comando POKE, poi di esaminare il valore e lasciarlo immutato. Il valore può poi essere usato come parte di qualche altro comando. Per esempio, il calcolatore trova i valori di posizione del paddle in posizione 54297 e 54298, così un'istruzione come:

PRINT PEEK(54297) **TAB**(20) **PEEK**(54298)

mostrerà questi valori sullo schermo. Questa combinazione di ordini indirizza il calcolatore in due tempi, uno dopo l'altro. I comandi PEEK mandano l'elaboratore a cercare quei valori che poi PRINT mostrerà sullo schermo.

Troverete molto utile il comando PEEK se prenderete confidenza con le posizioni di memoria usate dall'elaboratore per memorizzare ordini. Non appena viene inviato al calcolatore un comando da eseguire, cambiano i valori nelle posizioni di memoria. Con il comando PEEK potete vedere quali siano tali posizioni in ogni punto, o utilizzarle per comandare ulteriormente il calcolatore.

Alcune posizioni di memoria contengono due valori invece di uno. Per queste, è necessaria una forma più complessa del comando PEEK per selezionare un valore dall'altro. Queste forme richiedono una conoscenza dettagliata dell'organizzazione della memoria che non staremo a discutere qui.

Potete anche usare i comandi POKE e PEEK all'interno dei programmi, come fareste per altri comandi.

Il comando SYS

Un altro comando, SYS, indirizza il calcolatore a una specifica posizione di memoria dove trova le istruzioni che fanno agire i comandi BASIC. Per esempio, le istruzioni per il comando di tastiera RUN STOP-RESTORE iniziano in posizione A65E ed è possibile mandare il calcolatore lì, dove comincerà a svolgere queste istruzioni mediante il comando:

SYS A65E

Guidato da un comando SYS il calcolatore andrà alle istruzioni in ogni posizione della sua memoria. Ad alcune di queste vengono automaticamente memorizzate le piccole subroutine numeriche che fungono da ordini per i comandi BASIC. Potete aggiungere le vostre routine in linguaggio macchina alle altre e se fate ciò, ricordatevi che gli indirizzi marcano l'**inizio** di una rou-

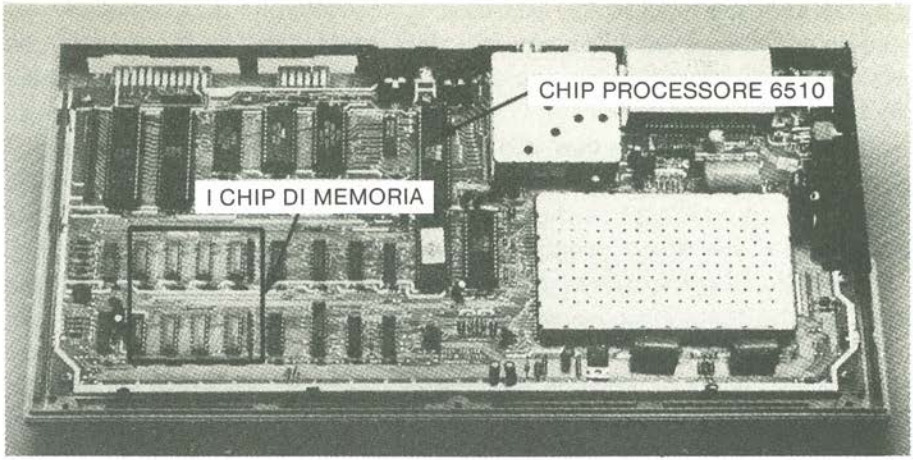

Figura 10.2 — L'elaboratore e la memoria del Commodore 64

tine. Se, con un'istruzione SYS, mandate il calcolatore a un indirizzo nel bel mezzo di una di queste sequenze, i risultati potrebbero essere distorti, anche se sarà possibile evitare tali sorprese dopo aver preso familiarità con queste posizioni.

COS'È IL COMMODORE 64?

Il Commodore 64 si basa sulla versione più recente di uno dei microcircuiti che hanno lanciato il personal computer. Il circuito che dà al Commodore 64 il suo carattere operativo è il processore 6510. Il 6510 è comandato dalle stesse istruzioni interne del suo predecessore, il 6502, un chip che si trovava nei primi calcolatori VIC-20, Apple II e Atari.

Di fatto, gli stessi programmi scritti nel linguaggio BASIC per il VIC-20 possono essere usati nel Commodore 64, purché non facciano riferimento (con comandi POKE, PEEK e SYS) a indirizzi di memoria diversi o a differenze di display (il VIC-20 infatti ha uno schermo di 22 colonne per 24 file). Inoltre, poiché il linguaggio BASIC nei diversi calcolatori consiste essenzialmente degli stessi elementi, i programmi scritti per una macchina possono essere a volte adattati ad altre.

In qualità di elaboratore e supervisore interno, il chip 6510 controlla il flusso di informazioni attraverso gli altri chip. Un unico pannello contiene la maggior parte dell'elettronica del Commodore 64. Qui risiedono altri otto chip, oltre al processore, come si può vedere in figura 10.2. Ognuno di que-

sti chip contiene 8192 (o 8K) posizioni di memoria, che possono essere occupate o vuote, a seconda delle istruzioni del processore. Questi otto chip insieme contengono la completa memoria del Commodore 64 consistente in 65 536 (o 64K) posizioni.

In questo calcolatore due altri chip eseguono lavori particolari. Uno, il chip di VIC, ha il compito di gestire grafici e illustrazioni programmate conosciuti col nome di "sprite". Questi sprite, una volta immagazzinati nella memoria del calcolatore, possono essere posti ovunque sullo schermo e fatti interagire. Un altro chip gestisce la produzione di suoni per mezzo del calcolatore; per suo tramite si possono creare musica sintetizzata ed effetti sonori.

L'utilizzazione di entrambi questi chip richiede un'elaborata programmazione delle posizioni di memoria o l'uso di programmi reperibili sul mercato in grado di tradurre comandi semplificati di tastiera, o movimenti di paddle o joystick, in modifiche delle posizioni di memoria.

COME ESPANDERE IL CALCOLATORE

Il vostro Commodore 64 non è stato terminato in fabbrica. Guardate le predisposizioni per i collegamenti e le prese sul retro e sul lato dell'alimentazione e vedrete. Potete terminarne la costruzione aggiungendo cartucce e inserendo cavi fino a che la macchina davanti a voi sia il più possibile aderente alle vostre esigenze.

Alcune cartucce — contenenti schede prodotte elettronicamente — possono modificare il modo di lavorare del vostro Commodore 64. Una è in grado di ammassare 80 caratteri di scrittura su uno schermo che ne poteva contenere solo 40. Un'altra cartuccia aggiunge un altro processore e il suo chip, innestando così un secondo calcolatore nel Commodore 64. Un'altra ancora trasmette informazioni e programmi dal calcolatore alle linee telefoniche e a destinatari molto lontani. Cavi e circuiti inviano il display dello schermo a stampanti, che trasformano le immagini in stampa su carta.

E, ovviamente, ci sono le cartucce dei giochi.

I dispositivi che aggiungete al calcolatore possono essere combinati per ottenere un sistema flessibile. La cartuccia di 80 colonne è un legame con **l'elaborazione delle parole,** un termine che significa semplicemente far muovere delle parole nella memoria del calcolatore, invece che su un foglio di carta. Un altro legame è la catena, una serie di istruzioni per comandi di elaborazione delle parole, molto opportunamente disponibili sotto forma di programmi reperibili sul mercato. Se le parole così elaborate devono eventualmente prendere forma come l'inchiostro sulla carta, alla fine della catena ci

dovrà essere una stampante. Il legame che completa il collegamento tra stampante e calcolatore è un cavo e, a seconda della stampante, un circuito extra connesso a quel cavo. Questi circuiti si comportano da tramite tra stampante e calcolatore.

Il Commodore 64 agisce ubbidendo alle istruzioni interne al suo chip 6510, prendendo comandi e programmi che "parlano" a quel chip. Un altro chip, il famoso Z80, è ben provvisto di programmi disponibili.

Inserendo una cartuccia contenente lo Z80 e i suoi circuiti, potete porre nel Commodore 64 un secondo elaboratore. Questo può utilizzare la memoria del Commodore 64 e ricevere comandi dalla tastiera o dal disco. Un Commodore 64, comunque, è la sua adattabilità alle possibilità offerte dai chip 6510 e lo Z80 per far girare programmi scritti per entrambi. In realtà voi commuterete tra due calcolatori in un'unica scatola. Il chip 6510 elabora comandi e informazioni in relazione alle istruzioni contenute nella memoria permanente del Commodore 64. Un'altra serie di istruzioni per il controllo dei processori (noto come "sistema operativo"), come lo Z80, viene sovente posta nei chip che la accettano, da un disco.

Questo sistema è conosciuto con l'acronimo CP/M® e, essendo uno strumento per controllare il funzionamento del processore, è molto noto. Di fatto, molti programmi scritti su macchine diverse che utilizzano il sistema operativo CP/M possono essere trasferiti da un tipo di calcolatore all'altro senza alcuna modifica, o quasi. Con una cartuccia Z80 inserita nel Commodore 64 è possibile caricare il CP/M nel calcolatore. Per il Commodore 64 il CP/M è prodotto sotto forma di una scheda contenente il chip Z80, assieme a un diskette.

Un'altra cartuccia, il modem, si inserisce nel Commodore 64 e forma un collegamento con le linee telefoniche. Con l'installazione di un modem, il vostro calcolatore si trasforma in una telescrivente elettronica, in grado di collegarvi ad altre persone e a banche dati in un modo che una volta era appannaggio esclusivo di grandi società e governi.

Forse la caratteristica più promettente di un personal computer come il Commodore 64 comunque è la sua adattabilità alle possibilità offerte dai nuovi strumenti. Il calcolatore è un mezzo per raggiungere un fine; una volta che abbiate imparato ad usarlo, potete utilizzarlo come un'estensione dei vostri stessi interessi e delle vostre idee. Non occorre che studiate l'elettronica o la logica dei chip che comandano la macchina per poterla utilizzare. Piuttosto aggiungerete abilità ad abilità per creare un sistema veramente "personale", comandato dalla vostra intelligenza.

APPENDICE A

COMANDI SPECIALI

Il vocabolario interno al calcolatore include comandi di utilità maggiore di quelli presentati nei capitoli precedenti. Questi comandi concernono principalmente la matematica e il trattamento dei numeri. Molti di questi sono raggruppati secondo la loro funzione. La sottolineatura indica quando nel comando potete (e solitamente dovete) includere un numero o una variabile.

Matematici e Trigonometrici

ABS (_): trasforma un numero negativo in positivo, lascia il numero positivo immutato.

ATN (_): Dà il valore trigonometrico arcotangente di un numero dato in radianti.

COS (_): Dà il valore trigonometrico coseno di un numero dato in radianti.

EXP (_): Dà il valore di **e**, 2.7182818, elevato alla potenza di un numero.

INT (_): Dà il valore di arrotondamento per difetto al numero intero più vicino.

LOG (_): Dà il logaritmo di un numero.

RND (_): Dà un valore casuale tra 0 e 1.0, senza riguardo al numero.

SGN (_): Dà il segno di un numero: –1 se il numero è negativo, +1 se positivo, 0 se 0.

SIN (_): Dà il valore trigonometrico seno di un numero dato in radianti.

SQR (_): Dà la radice quadrata di un numero.

TAN (_): Dà il valore trigonometrico tangente di un numero dato in radianti.

_^_: Eleva il numero che lo precede alla potenza del numero che segue.

_E_: Moltiplica il numero che lo precede per 10 alla potenza del numero che segue.

FN_(_): Dà un numero secondo la relazione matematica stabilita con un comando DEF. Vedi DEF FN.FN Z(A), FN Z(B) e FN Z(5), per esempio, ogni chiamata con la stessa relazione posta a FN Z(_).

DEF FN_(_): Definisce una funzione; stabilisce cioé un'equivalenza matematica tra la variabile in parentesi e la relazione matematica dopo il segno di uguale. Per esempio, DEF FNX(A) = A + B stabilisce una funzione FN X(_) che aggiunge il valore di B al valore o alla variabile in parentesi.

Confronti

Operatori di comparazione producono valori che corrispondono alla verità o falsità dell'affermazione fatta tramite loro. Quando l'affermazione è vera, viene dato un valore di −1; quando è falsa un valore 0. Come dice il loro nome, il tipo di affermazione fatta da questi operatori è sempre un confronto di valori.

_>_: (Maggiore di) Dà un valore di −1 se il numero che lo precede è maggiore di quello che segue, 0 negli altri casi.

_<_: (Minore di) Dà un valore di −1 se il numero che precede è minore di quello che segue, 0 negli altri casi.

_=_: (Uguale a) Dà un valore di −1 se il numero che precede è uguale a quello che segue, 0 negli altri casi.

_< >_ oppure _><_: (Disuguale) Dà un valore di −1 se il numero che precede non è uguale a quello che segue, 0 se lo è.

_>=_ oppure _=>_: (Maggiore di o uguale a) Dà un valore di −1 se il numero che lo precede è maggiore di o uguale al numero che segue, 0 se è minore.

_<=_ oppure _=<_: (Minore di o uguale a) Dà un valore di −1 se il numero che lo precede è minore o uguale al numero che segue, 0 se è maggiore.

Logica (Booleana)

AND: Dà un valore di 1 se il numero o la relazione che lo precede e quello che segue sono entrambi maggiori di 0, 0 se uno è 0.

_OR_: Dà un valore di 1 se il numero, la relazione che precede o quello che segue è 1 e l'altro è 1 o 0, 0 se entrambi sono 0.

NOT_: Dà un valore uguale al numero o alla relazione più 1, moltiplicato per –1.

Altri

ASC("_"): Dà un numero di codice, chiamato codice ASCII, per il primo carattere della stringa di caratteri.

CLR: "Clear" (pulisce) tutte le variabili di un programma sostituendo i relativi valori con 0.

STR$_: Ordina al calcolatore di trattare un numero come se fosse un carattere di stringa, invece che come valore numerico.

VAL_$: Ordina al calcolatore di trattare un carattere di stringa numerico come un numero.

APPENDICE B

GUIDA ALLE FRASI IN GERGO

Il gergo è la maledizione degli specialisti, ed ogni specialità ha il suo gergo. Lo incontrerete tra i dilettanti, i venditori e i programmatori. Coloro che lo parlano, e sono spesso ben informati, possono fornire informazioni utili altrimenti non ottenibili da coloro che parlano la lingua madre.

Questa guida vi può aiutare ad interpretare il loro linguaggio.

Applications program: Una serie di comandi, generalmente disponibile su disco che governa il calcolatore nell'esecuzione di determinati lavori i cui risultati compaiono in forma riconoscibile al di fuori della memoria del calcolatore.

Back-up: È una copia conservata per il caso che l'articolo possa venire a mancare. La back-up copy (copia di riserva) di un disco, oppure la back-up copy di un file o di un programma può essere conservata in un'altra forma o sullo stesso disco.

Bit: È il più piccolo frammento di informazione che il calcolatore possa gestire. Un bit ha sempre valore 0 o 1.

Boot a disk: È l'invio di istruzioni che fanno funzionare nel calcolatore un sistema disco da un disco in un'"unità".

Boot DOS: Vedi "Boot a disk".

Buffer: Una memoria controllata in modo speciale che conterrà dati o istruzioni che dovranno spostarsi tra il calcolatore e altre unità. È utile quando il flusso che esce è maggiore della capacità dell'unità che deve riceverlo, come se fosse un serbatoio che raccolga le acque di un fiume più in fretta di quanto non le lasci passare attraverso le porte di deflusso.

Bug: È una risposta non pianificata che interferisce con il funzionamento presunto della macchina. Il termine generalmente si riferisce al problema di un comando non progettato all'interno di un programma.

Byte: Il gruppo di informazioni-base inviate dal circuito del calcolatore in cui ciascuno degli otto bit ha valore 0 o 1. Ogni posizione di memoria nel Commodore 64 contiene un byte.

COBOL: Un linguaggio di programmazione di uso molto esteso fin da prima della comparsa dei personal computer, più piccoli.

Copy-protected: Un programma su disco che include le istruzioni progettate per impedire che quel programma venga copiato su un altro disco.

CPU: Central Processing Unit (Unità centrale di elaborazione), l'elaboratore del calcolatore.

Crush: Ciò che accade nello svolgimento di un programma quando il calcolatore smette all'improvviso di agire sui comandi di un programma, in maniera imprevista, generalmente a causa di istruzioni che il calcolatore non può eseguire.

Data-base Manager: Un programma che organizza le informazioni in accordo a un piano prestabilito.

Default: Il valore scelto automaticamente dal calcolatore quando questa scelta non viene fatta dall'utente del calcolatore stesso.

Enter: Invio di dati o istruzioni nel calcolatore, generalmente dalla tastiera, seguito dalla pressione del tasto RETURN.

Error: Qualsiasi dato o comando dato al calcolatore su cui esso non possa agire.

Escape Sequence: Uso di scrittura dalla tastiera per far arrestare lo svolgimento di un programma o parte di esso.

Fatal Crash: Un "crash" che si risolve nella perdita dalla memoria del calcolatore di comandi o di informazioni importanti.

Firmware: Quelle istruzioni contenute nella memoria che non possono essere modificate dalla tastiera (Cfr. "hardware" e "software").

Format: Incidere un dispositivo magnetizzabile così da consentire la memorizzazione di dati e programmi.

FORTRAN: Un linguaggio di programmazione usato da scienziati e tecnici fin dai tempi delle schede perforate.

Function: Un comando che manipola valori, spesso in modo non visibile. I comandi +, –, ∗, e / rappresentano le funzioni aritmetiche.

Hardware: Le parti materiali di un calcolatore, in opposizione alle istruzioni e ai valori ivi memorizzati (cfr. "firmware" e "software").

Hex: È l'abbreviazione di esadecimale (hexadecimal), un modo di rappresentare i numeri con un ibrido delle prime sei lettere (quindi hex) dell'alfabeto e dieci numerali (quindi decimal).

High-level Language: Una serie di comandi che controllano il calcolatore in modo relativamente complicato per svolgere un lavoro complesso me-

diante un unico comando. Il BASIC è un high-level language (linguaggio avanzato).

Housekeeping: Sono i lavori fatti dall'utente del calcolatore, o dal calcolatore stesso, per far funzionare un sistema o un programma.

Initialize: Formattare un disco ed aggiungere un programma iniziale di comandi che saranno svolti automaticamente dal calcolatore quando il disco viene scandito.

Instruction: Qualunque comando o disposizione dato al calcolatore.

Interface: Come nome, il circuito che serve come collegamento elettronico al calcolatore, o un sistema di comandi che collega diversi sistemi logici. Come verbo, collegare.

I/0: Flusso di dati o comandi da e per il calcolatore, Input/Output.

K: Quando ci si riferisce al numero di posizioni di memoria, la quantità equivale a circa un migliaio (il suo valore esatto è 1024). 64K equivale a 65 536.

Logo: Un linguaggio di programmazione creato per agevolare l'uso e l'apprendimento, un'introduzione all'uso dei calcolatori adatto ai bambini.

Loop: Una serie di comandi attraverso i quali il calcolatore viene fatto agire in maniera ripetitiva.

Low-level Language: Una serie di comandi che fanno agire il calcolatore in maniera relativamente semplice, e che richiede parecchi comandi per svolgere un lavoro complesso.

Machine Language: I codici di posizione di memoria propri di ogni calcolatore.

Menu: Non è una scelta di portate al ristorante, ma una lista di alternative offerte in un programma, fra cui l'utente sceglierà operazioni diverse. I programmi che se ne avvalgono sono detti **menu driven.**

Microcomputer: Un calcolatore costruito attorno a un microprocessore. Il Commodore 64 è un microcomputer.

Microprocessor: Il circuito elettronico che controlla il resto del calcolatore. Il chip 6510 è il microprocessore del calcolatore Commodore 64.

Mini-floppy Diskette: Il nome formale di un tipo di disco flessibile, del diametro di 5 1/4 pollici usato nel Commodore 64 e nella maggior parte degli altri personal computer.

MMU: (Memory Management Unit) La parte del calcolatore che organizza il flusso di informazioni nelle diverse parti della memoria seguendo uno schema prefissato.

Modem: Un dispositivo che accoppia il calcolatore alle linee telefoniche.

Parallel: Che lavorano simultaneamente, il modo in cui impulsi di dati e comandi possono essere inviati da un calcolatore attraverso una serie di cablaggi paralleli. (Contrario di **serial).**

Pascal: Un linguaggio di programmazione particolarmente diffuso negli ambienti accademici.

PC Board: (Printed Circuit Board) Un pannello in plastica rigida o in fibra su cui piste metalliche servono da conduttori elettrici in luogo dei cablaggi e su cui sono posti i componenti elettronici.

Peripheral: Qualsiasi dispositivo collegato al calcolatore, che lo controlli o che da esso sia controllato.

Pixel: Il punto più piccolo che un calcolatore sia in grado di controllare su uno schermo.

Powerful Language: Una serie di comandi, ognuno dei quali può dare istruzioni che, in un linguaggio "più debole", richiederebbero svariati comandi.

Powerful Program: Un programma in grado di dare molto con poche richieste o con un limitato controllo da parte dell'utente.

RAM: Sigla Random-Access Memory, le posizioni di memoria del calcolatore i cui valori possono venire modificati dalla tastiera.

Read: Trarre informazioni da una forma codificata, come per esempio un disco.

Return a Value: L'azione compiuta da un calcolatore quando fa comparire, sullo schermo o sulla stampante, numeri o caratteri in risposta a un comando.

ROM: Acronimo di Read-Only Memory, le posizioni di memoria di un calcolatore con valori fissati in fase di fabbricazione. Fornisce tutte le istruzioni interne al calcolatore.

Serial: Uno dopo l'altro, il modo in cui gli impulsi di dati e comandi possono essere inviati da un calcolatore attraverso un singolo cablaggio (opposto di **parallel**).

Software: Le istruzioni, come quelle disponibili in qualità di programmi su disco, che possono essere aggiunte e levate dal calcolatore e opposte alle sue parti materiali, fisse. (Cfr. con "firmware" e "hardware").

Spreadsheet: Un metodo visivo per organizzare ed effettuare calcoli su dati, professionali o finanziari, in colonne e file. Molti di questi sistemi sono disponibili su disco.

Utility: Un programma che comanda il calcolatore nello svolgimento di lavori interni, come quello di far muovere i dati nella sua memoria.

Write: Porre informazioni in modo tale che possano essere memorizzate e poi reperite.

Write-enable Notch: Una tacca sulla busta del disco che, se scoperta, consente all'unità disco di aggiungere e modificare le informazioni.

Write-protection: Un dispositivo del disco che evita che siano aggiunte o modificate le sue informazioni. L'assenza della tacca sulla busta del disco o la sua copertura danno la protezione contro la scrittura.

Questo libro vi insegnerà in poche ore ad usare il vostro Commodore 64, cominciando dalla tastiera e dal video, per passare poi alle altre periferiche più comuni, l'unità a dischi e il registratore a cassette.

Imparerete con estrema semplicità a scrivere programmi in BASIC, ma se questo non è il vostro obiettivo potete "saltare" i Capitoli dedicati alla programmazione e imparare invece come utilizzare "pacchetti" di software preconfezionati.

Potete imparare a:

— utilizzare le prestazioni grafiche del video;
— usare l'unità a dischi;
— scrivere programmi di simulazione;
— adattare alle vostre necessità programmi scritti da altri;
— espandere il sistema con diverse periferiche.

Lo stile è semplice e il linguaggio evita il più possibile termini gergali per essere accessibile a chiunque.

L. 13.500

Cod. 400D ISBN 88-7056-050-3

121

FACILE GUIDA AL COMMODORE 64

Joseph Kascmer

GRUPPO EDITORIALE JACKSON